AppleWorks 2.0 Program Documentation
LAN 2.1 Program Modification

TABLE OF CONTENTS

TABLE OF CONTENTS

Let's Talk: 06/17/82 07:25
Bob Lissner

Telephone: (702) 831-1722
On-line 24 hours

Welcome to this bulletin board for AppleWorks accessory developers.

Master Menu

1. INTRODUCTION
2. LITERATURE
3. ELECTRONIC.MAIL
4. QUIT

1. INTRODUCTION

This system runs at 1200 or 300 baud, depending on your modem.

This bulletin board system assumes that you are recording the information you receive. For that reason, and to reduce your phone bill, you don't get much of a chance to read the information while it is being sent to you.

If you want to stop and read each screen of data, you'll need to know the following commands:

Control-S or **Option-S** stops the text display

<SPACE-BAR> to resume the display

Control-X or **Option-X** stops the text display and returns to previous menu

This bulletin board was designed for software engineers, but is open to all callers. Your comments and questions are appreciated and should be sent to:

AppleWorks Product Manager
Claris Inc.
440 Clyde Ave.
Mountain View, CA 94043.

Read "Read.Me.First" in "Literature" for a summary of each document.

-- End of Introduction

2. LITERATURE

1. FILES.11.TO.20
2. A1
3. WHATS.NEW
4. FILES.1.TO.10
5. FILES.21.TO.30
6. FILES.31.TO.40
7. READ.ME.FIRST

H)elp E)xpand P)revious menu U)pload file M)aster menu

H)ELP

1. Menu.Choices: Choosing from our presentation of files

You can take a tour of the files offered in this presentation.

Some of these files allow comments to be added to them. You may choose to comment when prompted to do so.

Pressing the number corresponding to a file presents that file to you. Some selections may cause other menus to appear.

Pressing E for E)xpand displays the title line (a brief description) of each file within the current menu.

Pressing "P" for P)revious menu removes the current menu and takes you backwards through the menus you've chosen.

Pressing M for M)aster menu will return you to the main log-on log-off menu.

2. Commands: Helpful Commands to Control Display

<CONTROL> S pauses the text display

<SPACE-BAR> to resume the display

<CONTROL> X stops the text display and returns to previous menu

<CONTROL> C to signal that you are done entering text when prompted for unlimited text comments.

-- end of help

7. READ.ME.FIRST

Index of all documents on file. Changed 4/25/88.

This bulletin board contains information that should be useful to developers. File formats will assist anyone working with AppleWorks files on disk, and the remainder of the information will assist developers working on accessories that run while AppleWorks is also running.

These files include source code from version 2.0, and also from LAN version 2.1, recently announced by Claris and currently in Beta test. 2.1 source code is correct as of 4/25/88, but may be changed prior to availability of the product. LAN version 2.1 works only on LANs.

The differences are minor between the 2.0 and 2.1 LAN source code provided here, as every attempt was made to keep variables, jump tables, and code segments in constant locations.

If you have suggestions about how we can improve the bulletin board, write or call the AppleWorks product managers at Claris. Your notes left on this bulletin board ARE NOT answered.

This bulletin board is not provided by Claris. No-one is guaranteeing that it's 100% right.

Future versions of AppleWorks may cause substantial changes in the contents of these files, but we do attempt to make as few changes as possible in the basic locations.

This bulletin board is running on an Apple //e with an 800KB disk, and Let's Talk software from Russ Systems, in Santa Cruz CA.

This information may be copied as necessary.

Please record this file. It is your only index to all the other files in this library.

INDEX OF FILES

Data base Common. File 1. Definitions used in all data base segments, including location of files and work areas in memory. 8K bytes. File 31 for 2.1 LAN.

File formats. Files 2 and 15. AppleWorks file formats on disk. 33K bytes. Unchanged for 2.1 LAN.

Main work area. File 3. Definitions used throughout Apple-Works. Defines common work areas used by all applications. Defines many of the symbols referenced in other files. 20K bytes. File 21 for 2.1 LAN.

File 3 contains most of the basic definitions.

Hooks documentation. File 4. Defines how to use "hooks" built into version 2.0 and 2.1, that allow you to intercept keystrokes and output to the screen. 3K bytes.

Host documentation. File 5. Documentation of routines in the permanently resident portion of AppleWorks. You can use these routines, at your own risk, of course, for a number of the functions that are basic to any assembly language program. File 6 defines where the routines are located. 6K bytes.

Host entry points. File 6. This defines addresses that can be JSR'd to for permanent routines. File 5 describes the functionality. 2K bytes. File 22 for 2.1 LAN.

Sub-host documentation. File 7. This describes more routines that are permanently resident. Most pertain to memory management. Please note that a patent is pending on these memory management routines, so you may end up paying us royalties if they're copied. 5K bytes.

Sub-host entry points. File 8. This file defines the entry points for sub-host routines. 1K bytes. File 23 for 2.1 LAN.

Spreadsheet Common. File 9. Definitions used in all spreadsheet segments, including location of files and work areas in memory. 16K bytes. File 38 for 2.1 LAN.

Word processor Common. File 10. Definitions used in all word processor segments, including location of files and work areas in memory. 10K bytes. File 36 for 2.1 LAN.

Seg.pr. Source code. Files 11, 12 and 13. 41K bytes. File 24 contains only changes in 2.1 LAN.

Memory manager. Use of //gs memory manager by version 2.0 and version 2.1. File 14. 2K bytes.

Bugs in 2.0, and conflicts between 2.0 and other programs being developed for the //gs. File 16. 2K bytes.

Seg.pr. File used to define parts of seg.pr as needed at various points within AppleWorks. 2.0 and 2.1 LAN. File 26.

Desktop manager entry points and variables for main segment. This is 2.1 LAN source, and is probably unchanged from 2.0. File 27.

Data base entry points and variables for main segment. This is 2.1 LAN source, and is probably unchanged from 2.0. File 32.

Data base report writer (print) common work area. 2.0 and 2.1 are the same. File 33.

Data base report writer main segment entry points. 2.0 and 2.1 File 34.

Data base additional common work area. 2.0 and 2.1. File 35.

Word processor main segment entry points and some variables. File 37 is 2.1 LAN. 2.0 is probably the same.

Spreadsheet main overlay entry points and some variables. File 39 is 2.1 LAN.

Spreadsheet main segment entry points and some variables. File 40 is 2.1 LAN.

-- End of index file

FILE CONTENTS EXPANDED

4. FILES.1.TO.10

```
1. FILE.1       ; Equates and zero page locations for Data base
2. FILE.2       ; Part I of File formats for AppleWorks and /// E-Z Pieces
3. FILE.3       ; Equates and definitions used throughout AppleWorks.
4. FILE.4       ; This is documentation of hooks in 2.0 Aplworks.system.
5. FILE.5       ; Documentation of permanently resident AppleWorks routines.
6. FILE.6       ; These are references to the routines in Host
7. FILE.7       ; Book SubHost contains many of the host routines.
8. FILE.8       ; These are references to the routines in SubHost
9. FILE.9       ; Common routines for SpreadSheet assembly
0. FILE.10      ; Common routines for Word Processor assembly
```

1. FILES.11.TO.20

```
1. FILE.12      ; Part II of seg.pr source code.
2. FILE.13      ; Part III of seg.pr source code.
3. FILE.14      ; How AppleWorks utilizes //gs memory manager
4. FILE.15      ; Part II of File formats for AppleWorks and /// E-Z Pieces
5. FILE.16      ; Bugs and conflicts in AppleWorks version 2.0.
6. FILE.11      ; This becomes segment Seg.PR.  It is definitions of printers.
```

5. FILES.21.TO.30

```
1. FILE.26      ;
2. FILE.23      ; These are references to the routines in SubHost.
3. FILE.24      ; This becomes segment Seg.PR.  It is definitions of printers.
4. FILE.21      ; Equates and definitions used throughout AppleWorks.
5. FILE.22      ; These are references to the routines in Host.
6. FILE.27      ;
```

6. FILES.31.TO.40

```
1. FILE.31      ; Equates and zero page locations for Data Base.
2. FILE.32      ; These are the entry points into Main Segment.
3. FILE.33      ; Zero page locations for RWSeg.
4. FILE.34      ; Book RWSegRef.
5. FILE.35      ; THE FOLLOWING MUST BE IN ORDER, AS THEY COME FROM DISK *.
6. FILE.36      ; Common routines for Word Processor assembly.
7. FILE.37      ; ClearFileHeader .Equ    LWPMain+2+3.
8. FILE.38      ; Common routines for SpreadSheet assembly.
9. FILE.39      ; Book of references to Main portion of Calculator Edit.
0. FILE.40      ; AccumType       .Equ           LSSMain+2+3.
```

--- End of file contents expanded

3. WHATS.NEW

What's new on the bulletin board:

4/25/88 Version 2.1 LAN files added. Additional files, not previously available on this bulletin board, are shown. Version 2.1 LAN is in Beta test, and is subject to change.

1/8/87 Warning of possible future change added to File 4, hooks documentation.

1/8/87 File 16, bugs, has been added to.

12/12/86 File 16 added. Bugs in 2.0, or conflicts between 2.0 and other programs being developed for the //gs.

11/20/86 Explanation of how AppleWorks uses the memory manager on the gs. File 14.

11/20/86 File formats were incomplete. Now Files 2 and 15.

11/20/86 File 2 changed to show that byte +92 of the WP header record pertains to mail-merge.

11/14/86 Source code for seg.pr. Files 11, 12 and 13.

-- End of What's new

--- FILE.3 ---

Main Work Area

```
; Equates and definitions used throughout AppleWorks.

; This file is included in each AppleWorks segment.  Other
; developers may be interested in the individual components
; $600 byte work area defined here.

;
; Parameters that control assembly.  ; Combined refers to whether
; the segments are combined into one file.  They are for distribution,
; are not for my testing.

Combined        .Equ            0               ; 1 or 0, boolean
Crippled        .Equ            0
Language        .Equ            "A"             ; Controls assembly
Opsys           .Equ            2               ; Controls assembly
Testing         .Equ            0
WarnEm          .Equ            0               ; 1 or 0, warn msg on splash

OAKey           .Equ            80              ; Open apple adds this
DateCode        .Equ            192.

;               Environment record information
EREL ength      .Equ            36.             ; Length of ER Printer entry
EROFormFeed     .Equ            80
EROPageStop     .Equ            40
EroNeedLF       .Equ            20

False           .Equ            0

;               Bits to tell WriteVBar (VBarWrRtn) to write relative
;               to top left of box
FCLeftSide      .Equ            80
FCTopLine       .Equ            80

Indent          .Equ            80              ; Flags Indent needed
LPrFixed        .Equ            136.            ; Fixed area in printer defns.
MaxOnDesk       .Equ            12.
OldCode         .Equ            207.
PosSegs         .Equ            43.
PosSegsX4       .Equ            172.
ScreenWidth     .Equ            80.
TimeCode        .Equ            212.
True            .Equ            1
TurnOnAStack    .Equ            0C009
TurnOfAStack    .Equ            0C008
ZP              .Equ            0               ; For Mac conversion

; Items that control assembly for foreign languages

                .If             Language="A"
KCommaInNo      .Equ            ","
KDateSep        .Equ            "/"
KDecPoint       .Equ            "."
KCurrency       .Equ            "$"
KVertBar        .Equ            "|"
No              .Equ            "N"
Yes             .Equ            "Y"
ComAlt1Help     .Equ            "/"+OAKey
ComAlt2Help     .Equ            "/"+OAKey
ComAlt3Help     .Equ            "/"+OAKey
ComClr2End      .Equ            25.             ; Control-Y
ComEditMode     .Equ            "E"+OAKey
ComEndSC        .Equ            "^"             ; To end printer special codes
ComHelp         .Equ            "?"+OAKey
ComHardCopy     .Equ            "H"+OAKey
ComQuick        .Equ            "Q"+OAKey
ComSave         .Equ            "S"+OAKey
LastLCChar      .Equ            "z"
LastUCChar      .Equ            "Z"
StdPL           .Equ            110.            ; Standard paper length
KSPMarker       .Equ            "^"             ; To display WP special functions.
                .Endc
```

```
                .If             Language="F"
KCommaInNo      .Equ            "."
KDateSep        .Equ            "/"
KDecPoint       .Equ            ","
KCurrency       .Equ            "£"
KVertBar        .Equ            "¦"
No              .Equ            "N"
Yes             .Equ            "O"
ComAlt1Help     .Equ            "/"+0AKey
ComAlt2Help     .Equ            "."+0AKey
ComAlt3Help     .Equ            "<"+0AKey
ComClr2End      .Equ            25.             ; Control-Y
ComEditMode     .Equ            "A"+0AKey
ComEndSC        .Equ            "^"+0AKey       ; To end printer special codes
ComHelp         .Equ            "?"+0AKey
ComHardCopy     .Equ            "H"+0AKey
ComQuick        .Equ            "Q"+0AKey
ComSave         .Equ            "S"+0AKey
LastLCChar      .Equ            "z"
LastUCChar      .Equ            "Z"
StdPL           .Equ            117.            ; Standard paper length
KSFMarker       .Equ            "\"             ; To display WP special functions.
                .Endc

                .If             Language="G"
KCommaInNo      .Equ            ","
KDateSep        .Equ            "."
KDecPoint       .Equ            ","
KCurrency       .Equ            "d"
KVertBar        .Equ            "¦"
No              .Equ            "N"
Yes             .Equ            "J"
ComAlt1Help     .Equ            "/"+0AKey
ComAlt2Help     .Equ            "~"+0AKey
ComAlt3Help     .Equ            "/"+0AKey
ComAltHelp      .Equ            7E+0AKey
ComClr2End      .Equ            25.             ; Control-Y
ComEditMode     .Equ            "C"+0AKey
ComEndSC        .Equ            "^"+0AKey       ; To end printer special codes
ComHelp         .Equ            "?"+0AKey
ComHardCopy     .Equ            "X"+0AKey
ComQuick        .Equ            "W"+0AKey
ComSave         .Equ            "S"+0AKey
LastLCChar      .Equ            "j"
LastUCChar      .Equ            "J"
StdPL           .Equ            120.            ; Standard paper length
KSFMarker       .Equ            "^"             ; To display WP special functions.
                .Endc


                .If             Language="I"
KCommaInNo      .Equ            "."
KDateSep        .Equ            "/"
KDecPoint       .Equ            ","
KCurrency       .Equ            "#"
KVertBar        .Equ            "¦"
No              .Equ            "N"
Yes             .Equ            "S"
ComClr2End      .Equ            25.             ; Control-Y
ComAlt1Help     .Equ            "/"+0AKey
ComAlt2Help     .Equ            ","+0AKey
ComAlt3Help     .Equ            "/"+0AKey
ComEditMode     .Equ            "X"+0AKey
ComEndSC        .Equ            "^"+0AKey       ; To end printer special codes
ComHelp         .Equ            "?"+0AKey
ComHardCopy     .Equ            "H"+0AKey
ComQuick        .Equ            "Q"+0AKey
ComSave         .Equ            "A"+0AKey
LastLCChar      .Equ            7E
LastUCChar      .Equ            "Z"
StdPL           .Equ            120.            ; Standard paper length
KSFMarker       .Equ            "^"             ; To display WP special functions.
                .Endc
```

```
; Hor and vertical for console driver.  Used in SubHost too
CH              .Equ            14              ; Position rel to top left
CV              .Equ            15              ; Position rel to top left
KOAChar         .Equ            81
KMouseReturn    .Equ            8D


;               First the items that are common between all files
FileBuf         .Equ            0800
HostWA          .Equ            0A00
LHostWA         .Equ            600             ; Length
HostLocn        .Equ            1000
LSanE           .Equ            2100            ; Apple SANE numerics package
LEElem          .Equ            0D000           ; Apple elementary functions
LHiSane         .Equ            21
LHiElems        .Equ            0D0
LZPSane         .Equ            0CC             ; Beg of 34 zero page locns.
LHiElem         .Equ            0D0
LSubHost        .Equ            0D000

DDWA            .Equ            7500            ; For disk directory work
SSWA            .Equ            7D00
DBWA            .Equ            7E00
WPWA            .Equ            7B00

; Equates for relative addresses of main modules
LOrgMain        .Equ            2100
LQCMain         .Equ            7500
LQFMain         .Equ            6E00
LQWMain         .Equ            4E00
LHiOrgMain      .Equ            21
LHiQCMain       .Equ            75
LHiQFMain       .Equ            6E
LHiQWMain       .Equ            4E


; Equates for relative addresses of segments for each system
LOrgSegs        .Equ            4000            ; 1.3 changed from 6300
LQCSegs         .Equ            3900
LQFSegs         .Equ            3900
LQWSegs         .Equ            2100

LQCOverlay      .Equ            5100

LHiOrgSegs      .Equ            40              ; 1.3 changed from 63
LHiQCSegs       .Equ            39
LHiQFSegs       .Equ            39
LHiQWSegs       .Equ            21

LHiQCOverlay    .Equ            51


;               For //e bank addressing.  Values temp room for SubHost
IOBuffer        .Equ            0BB00

PDDevCntMl      .Equ            0BF31           ; ProDOS devices attached
PDDevLst        .Equ            0BF32           ; ProDOS devices attached
IIMainBase      .Equ            9F00            ; First storage avail in main
IIHiMainBase    .Equ            9F
IIAuxBase       .Equ            0800            ; First storage avail in aux.
IIHiAuxBase     .Equ            8

MaxQCRows       .Equ            999.
RdMain48        .EQU            0C002
RdAux48         .EQU            0C003
WrMain48        .EQU            0C004
WrAux48         .EQU            0C005
RWAux16         .EQU            0C009
RWMain16        .EQU            0C008
```

```
; Type codes for numeric package
FFExt           .Equ            0
FFDbl           .Equ            1
FFInt           .Equ            4


; Op codes for numerics package
FOAdd           .Equ            0
FOSetEnv        .Equ            1
FOSub           .Equ            2
FOGetEnv        .Equ            3
FOMult          .Equ            4
FODiv           .Equ            6

FOD2B           .Equ            9
FOB2D           .Equ            0B

FOSQRT          .Equ            12
FORTI           .Equ            14              ; Round to integer
FOTTI           .Equ            16              ; Truncate to integer
FOClass         .Equ            1C
FOABS           .Equ            0F              ; Absolute
FOCMP           .Equ            08              ; Compare
FONeg           .Equ            0D
FOZ2X           .Equ            0E
FOX2Z           .Equ            10
LExtNum         .Equ            10.

; And opcodes for Elems package
FOXPwrY         .Equ            12

; Application call codes.  Host will place these values in X
; when making calls to the individual Main routines.  A high bit added
; means to just get the proper modules loaded for this, but not called.
;                                 Returns to
ACCloseFile     .Equ            1   ; Caller, put back in banks, release WA
ACInitFile      .Equ            2   ; Caller, is in WA, needs pointers inited.
ACMergeFile     .Equ            83  ; Host,
ACNewFrKB       .Equ            84  ; Host,
ACNewFrCB       .Equ            85  ; Host, Clipboard
ACOpenFile      .Equ            6   ; Caller, Gets from banks, does init
ACReleaseFile   .Equ            7   ; Caller, release the bank switched
ACResumeFile    .Equ            88  ; Host, already in work area, init done

IOFailed        .Equ    ZP+003

;               All of the following depend on whether or not Slinky
;               is working.
ALVDBMMP        .Equ    ZP+004      ; A(Ptr for maximum record)
ALVWPMMP        .Equ    ZP+006      ; A(Ptr for maximum line)
AEndSSWA        .Equ    ZP+008      ; A(Last byte of BldRecWA)
MaxDBRecs       .Equ    ZP+00A      ; Maximum recs allowed in DB
MaxWPLines      .Equ    ZP+00C      ; Maximum lines allowed in WP
MaxSSRSize      .Equ    ZP+00E      ; Maximum record size in SS

AuxReg          .Equ    ZP+080
CDAddr          .Equ    ZP+080      ; Used by console driver
CurHor          .Equ    ZP+082      ; Cursor position
CurVer          .Equ    ZP+083      ; Cursor position 0..21
CWork1          .Equ    ZP+084
SpaceOK         .Equ    ZP+085      ; Cleared, set by SMPut routines.
FoundCommand    .Equ    ZP+086
FoundReturn     .Equ    ZP+087
FoundEscape     .Equ    ZP+088      ; Non zero if escape found
HelpAvail       .Equ    ZP+089
I               .Equ    ZP+08A
IntFrKB         .Equ    ZP+08C      ; numeric
LLR             .Equ    ZP+08D
LLB             .Equ    ZP+08F      ; is a word

; Work area for Multiply
MReg            .Equ    ZP+091
MWrkX           .Equ    ZP+093
MWrkY           .Equ    ZP+094
PageBegin       .Equ    ZP+095
PageEnd         .Equ    ZP+096
ReturnCode      .Equ    ZP+097
```

```
; Return 0 through ZReg02 must be in order
RETURN0         .EQU    ZP+098
RETURN2         .EQU    ZP+09A
AArg            .Equ    ZP+09A
RETURN4         .EQU    ZP+09C
BArg            .Equ    ZP+09C
ZReg00          .Equ    ZP+09E
CArg            .Equ    ZP+09E
ZReg02          .Equ    ZP+0A0
DArg            .Equ    ZP+0A0
R1              .Equ    ZP+0A2
ExitFlag        .Equ    ZP+0A4      ; HAS 0A-0 or 0A-5 that caused it
                                    ; to bail out of the current file
Work1           .Equ    ZP+0A5
FR              .EQU    ZP+0A6
FBR             .Equ    ZP+0D0
;

; These are output to the screen codes; For My ConsDriver
ClearELine      .Equ    1
ClearLine       .Equ    2
ClearVP         .Equ    3
ClearEVP        .Equ    4
AbsPosn         .Equ    5
Left            .Equ    6
Right           .Equ    7
Up              .Equ    8
Down            .Equ    9
Inverse         .Equ    10.
Normal          .Equ    11.
VPBot           .Equ    12.
Return          .Equ    13.
VPTop           .Equ    14.
VPReset         .Equ    15.
ResetVP         .Equ    15.
Bell            .Equ    16.
HorShiftVP      .Equ    17.


; These are input codes, some output too.
ComCurLeft      .EQU    8
ComCurRight     .EQU    21.
ComCurUp        .EQU    11.
ComCurDown      .Equ    10.
ComTabLeft      .Equ    137.
ComTabRight     .Equ    9.
ComEscape       .EQU    27.
ComDelKey       .Equ    7F
ComReturn       .Equ    13.

ComLineFeed     .Equ    10.

ComFormFeed     .Equ    0C

Buffer          .Equ            HostWA
```

```
;                All the stuff for the clipboard.  This must be in order
;                as it is cleared as one block of 515 bytes.
KlipBoard        .Equ            Buffer+80.

; KBHigest is not consistant between applications:
; WP:  The count of items on the clipboard
; DB:  The highest place used, values from 0 - 255. (0: 1 place used)
KBHighest        .Equ            KlipBoard+512.
KBType           .Equ            KBHighest+2     ; Type of data L C T or M
;                End of what must be in order

CurrFNo          .Equ            KBType+1

;                Stuff for organizer, 12 items on desk.
DTCurOpen        .Equ            CurrFNo+1
DTCtOnDesk       .Equ            DTCurOpen+1

;                The file that is actually open on the desk, 32 bytes
DTFName          .Equ            DTCtOnDesk+1
DTFType          .Equ            DTFName+21.
DTFStatus        .Equ            DTFType+1
DTFPicked        .Equ            DTFStatus+1     ; Boolean:  Picked for save
DTFSizeK         .Equ            DTFPicked+1     ; 1.3 changed to 2 bytes
DTFAddress       .Equ            DTFSizeK+2      ; Bank address if not curr.


;                Pointers to the bank switched places where the up to
;                12 files are stored.  When one is removed, all higher
;                are moved down so that the n that are in, are contiguous.
;                Leave the first unused, so that accum will point right
DTFPointers      .Equ            DTFName+32.

FieldName        .Equ            DTFPointers+26.
FoundSpace       .Equ            FieldName+21.
HardPathName     .Equ            FoundSpace+1
;                Prodos DSSSO000 device number.  If non zero, use this
;                to find a file catalog SOS value 01 to 06, location of
;                file cat.
MainPathName     .Equ            HardPathName+1
                                 ; If HardPath is 0, then use this
                                 ; Pathname.  If Hard <> 0, then
                                 ; this is just for display

; Where we keep the stack of routine names
CurrMode         .Equ            MainPathName+32.
StackP1          .Equ            CurrMode+21.
StackP2          .Equ            StackP1+21.
StackP8          .Equ            StackP2+126.

NewFile          .Equ            StackP8+21.
Newstr           .Equ            NewFile+1
OldStr           .Equ            NewStr+128.     ; Max length 127.
OneChar          .Equ            OldStr+128.
OrgInMem         .Equ            OneChar+1       ; Organizer in memory
RacPDNo          .Equ            OrgInMem+1
R2               .Equ            RacPDNo+1
R3               .Equ            R2+2

SavedHor         .Equ            R3+2            ; MUST be Hor, then ver
SavedVer         .Equ            SavedHor+1
SegNum           .Equ            SavedVer+1      ; SOS only segment number
SMAddr0          .Equ            SegNum+1        ;
WVVAddr          .Equ            SMAddr0+2
WVVCount         .Equ            WVVAddr+2
VertTable        .Equ            WVVCount+2      ; X, Y, Length, entries 0-30
VTHighest        .Equ            VertTable+93.   ; Highest entry in table
Work2            .Equ            VTHighest+1
Work3            .Equ            Work2+2
```

```
; Environment record has starting file cabinet, and printer specifications
; IF ANY OF THIS CHANGES, YOU HAVE TO CHANGE SEGPR, WHICH CONTAINS
; THE INITIAL VALUES FOR ALL OF THIS.
EnvRec                  .Equ            Work3+3
ERHardPathName          .Equ            EnvRec
ERMainPathName          .Equ            ERHardPathName+1
ERPtrCount              .Equ            ERMainPathName+32.      ; 0-3
ERDAWPrinter            .Equ            ERPtrCount+1            ; 0-3
ERMainPrinter           .Equ            ERDAWPrinter+1          ; 0-3
ERCurrPrinter           .Equ            ERMainPrinter+1
ERToday                 .Equ            ERCurrPrinter+1
ERFiller1               .Equ            ERToday+9 ; String Normally [I] BON

; The following is repeated three times
ERPtrName               .Equ            ERFiller1+10.
ERDriverName            .Equ            ERPtrName+16.    ; Nada on //e, codes
                                                         ; to init port on //c
; Next says spec rules for //c (c), //e (e), or on disk (d)
ERSpecOn                .Equ            ERDriverName+15. ; Where custom made.
ERDeviceNo              .Equ            ERSpecOn+1       ; Or slot or FF (disk)
ERPtrType               .Equ            ERDeviceNo+1
EROptions               .Equ            ERPtrType+1      ; Boolean
ERPlaten                .Equ            EROptions+1

EnvRecEnd               .Equ            ERPtrName+108.
; End of environment record

; So other segments can get past the environment record on disk
EnvRecLength            .Equ            EnvRecEnd-EnvRec
D100Length              .Equ            300             ; Space saved in Seg.pr

; Variables that specify current (Organizer) file card, and position
; of its top left corner
DFCNumber       .Equ    EnvRecEnd
DFCHorPosn      .Equ    DFCNumber+1     ; Hor posn of left side
                                        ; Includes preceding 2 sp.
DFCVerPosn      .Equ    DFCHorPosn+1    ; Vert posn of top left byte

;               Boolean whether code is coming from Profile
CodeOnFive      .Equ    DFCVerPosn+1    ; Initial is zero (false)
ClockByte       .Equ    CodeOnFive+1    ; Byte at 0BF06 (ProDOS clock)
CSWValue        .Equ    ClockByte+1     ; 2 bytes, value of 36,37 at entry
Clearing        .Equ    CSWValue+2      ; Boolean, ReadKB is for ClearTA

PrSlots         .Equ    Clearing+1      ; //e, slots 0-7 valid for pr. bool
DidSwapDisk     .Equ    PrSlots+8       ; Boolean, data disk in drive 1


FloppyList      .Equ    DidSwapDisk+1 ; For ProDOS, list of drives, like 60,
E0.
                                        ; For SOS, first byte is highest drive #,
                                        ; like X'02', and remainder is not used

LastDBPrinter   .Equ    FloppyList+6
LastWPPrinter   .Equ    LastDBPrinter+1
LastSSPrinter   .Equ    LastWPPrinter+1

Today           .Equ    LastSSPrinter+1
SlinkAddr       .Equ    Today+20.       ; Value for X reg., from BFF8
SlinkSlot       .Equ    SlinkAddr+1

ConsDevNum      .Equ    SlinkSlot+1     ; PROBABLY AVAILABLE.
IsLolly         .Equ    ConsDevNum+1

; RamDisk driver address and device number
RamDiskD        .Equ    IsLolly+1
RamDiskN        .Equ    RamDiskD+2
```

```
; Routine for when desktop is full
DontBitch       .Equ    RamDiskN+1

PEChar          .Equ    DontBitch+1     ; Char for bottom line.
KDTAvail        .Equ    PEChar+1        ; 1.3 Word: Last K bytes available.
SlinkMin        .Equ    KDTAvail+2      ; 1.3 Size of slinky blocks.
SlinkShifts     .Equ    SlinkMin+1      ; 1.3 How many times to rotate on Slinky
StrWork5        .Equ    SlinkShifts+1   ; 1.3 6 bytes from Word2Str
KBWidth         .Equ    StrWork5+6      ; 80 only so far.
MyID            .Equ    KBWidth+1       ; 2 bytes, Cortland
Seg00Type       .Equ    MyID+2          ; 1 byte, which Seg00 loaded

; Typeahead for Apple //e
TAArea          .Equ    Seg00Type+1
TALength        .Equ    24.
TANextIn        .Equ    TAArea+TALength
TANextOut       .Equ    TANextIn+1

-- End of file.3
```

Host Documentation

; Documentation of permanently resident AppleWorks routines.

CALLSEG

; Routine to load segments. READ IT, you'll like it. It keeps track of
; which Main is in, and whether or not Organizer is in. As segments are
; loaded from disk, it tries to store a copy in bank-switched. If still in
; bank-switched at the next call of this seg, then the copy will come from
; bank-switched. Mucho faster, of course.
;
; Also of interest is that this works differently for my testing, as opposed
; to the distributed software. Testing reads individual segments from disk.
; Distribution reads segments 01-43 from a single file, seg.mn. Location in
; seg.mn is a series of three bytes pointers at the beginning of seg.mn. See
; program /p2/myasm/combine for how this is built.

CC2S

; Routine to concatenate two strings. First string has second added at end.

CLEARDA

; Routine to clear the window within the display area. Clears from PageBegin
; to PageEnd

CLEARTA

; Routine to check the typeahead and see if anything waiting. Sets switches
; including FoundEscape,FoundSpace and FoundReturn.

CLEARWINDOW

; Routine to clear a specific window. Clears all columns from row X to row Y.

CountTA

; Counts the number of characters in TypeAhead. Returns true or false in
; accum, whether there are any. Book Host2. Console driver.

```
ConsDriver

; Puts (accum)  characters, pointed at by CDAddr, out to the console.  If
; characters are less than space (x'20') they are control characters.  Look
; in the Device drivers manual for what they do.  Note that the Equates are
; different for the //e.

Conv1TP

; Routine to convert accumulator to printable 3 digit number with prec blanks.
; Result goes into accum, X, Y, and into StrWork3.  Preceding zeroes are
; changed to blanks.

DieNow

; Locks up the system, and stores error information.

DoBell

; Rings the bell.  Returns condition code of non-zero (BT will pick it up).

DoGoToXY

; Moves the cursor to column X, row Y on screen.

DoHelp

; Writes OA-? for help on bottom of screen.

EnterCommand

; Puts something like "Enter your command" on bottom line, left side.

FullExit

; FullExit calls Organizer.  Get here at beginning, and also returning from
; "Main" routines.  Have to come here because Organizer may not be in memory
; at the conclusion of the Main.  Some "Main" calls return directly to the
; caller in Organizer.

GetScrLine

; Routine gets a line from the screen.  Input: accum has line number.
; Output, 80 characters at FileBuf+256 thru FileBuf+256+79.
```

```
GetArgs

; Get a variable number of arguements from bytes past the JSR.  Accum will
; have number of bytes to pull.  Updates the return address by number of
; bytes. The pulled bytes go to AArg, AArg+1, etc.

GetVBar

; GetVBar lets the user pick an item from the numbered items now on the
; screen.  The user can move the number bar by using the up and down arrows.
; Accum at entry has the starting number.  It will almost always be 1, unless
; the program is smart enough to remember what should be used.  Returns in
; accum the number picked, or zero if escape was pressed

HighLight

; HighLight inverses characters that are already on the screen.  This is used
; by deletes, moves, etc.  x and y have screen posn, Accum has number of
; bytes.  Accum may be zero, which will just remove all inverse from the
; line.  Always does whole line.

IODisable

; For the //e only.  Prevents IO.  Not needed on MAC.

IOEnable

; For the //e only.  Allows disk IO.  Not neededd on MAC.

MakeIB

; Clears the screen, writes top and bottom solid lines , file name, escape
; road map.

MadeAChange

; MadeAChange posts the changed flag on current file.

MultByte

; Actual multiply routine for register X times register Y into  MREG (two
; bytes)

MultWord

; Routine to multiply two words.  Addresses follow the JSR MultWord.  Result
; in MReg (4 bytes)
```

```
MvLeftRtn

; Moves bytes from left to right.
; First word following JSR:  To
;                Next word:  From
;                Next word:  Number of bytes


MvRightRtn

; Same as MvLeftRtn, but moves from right to left.


PopStack

; Removes current item from the Escape road map.  Displays it.


PleaseEnter

; Puts something like "Please type your selection: on line 23.


PressAny

; Routine displays "Press any key to continue".


Print

; Print puts (accum) chars to the printer in slot 1.  Address of data
; (not a string) will be in word following JSR.
; 00 in accum:  Open the printer.
; FE in accum:  Output a carriage return, line feed.
; FF in accum:  Close the printer.


PushStack

; Routine to push the escape roadmap, adding a new string to it, arguement is
; new string that goes into the stack


ReadKB

; Gets one byte from the keyboard.  Leave it in accum.


RestCursor

; Routine to restore cursor position to wherever it was after the last
; WriteCom.  SaveGoToXY can also set this.


SaveGoToXY

; Moves cursor to X,Y, and saves these values for future use by RestCursor.
```

```
SetUCC

; Convert accumulator to upper case, not too tough.

SetUCS

; Convert a string to upper case.

StrCmpRtn

; Routine to compare two strings.

StrMvRtn        LDA             #4

; Routine to move one string to another.

StrWrRtn

; Routine to write a string at coordinates given.
; Return2 has Hor and Vert, Return4 has string address.

TestKB

; Routine checks keyboard.  If it finds anything, places in typeahead.  Would
; like to ring bell when full, but this would be recursive and kill me.

Wait

; Routine to wait (accum) tenths of a second.  Just sit.

WipeOutTA

; Routine to clear the type-ahead.

WritiByte

; Write character in register X at current cursor position.  Move the cursor
; one to the right after.

WriteCom

; Writes the string at 0,23.  Saves the next available cursor position for
; future use by RestCursor.
```

```
VBarWrRtn

; Writes a string at the given cursor position.  Numbers it, and displays the
; number ahead of it.  Saves location, length, and number for subsequent use
; by GetVBar.


Write

; Writes the string at wherever the cursor is now.


WritProp

; Writes (register Y) character on screen (register x) times.
; Examples:    -                                        80


WriteRUSure

; Write "Are you sure"  Question mark will be placed by GetYN.


-- End of file.5
```

Host Entry Points

```
; These are references to the routines in Host
; Skip          JMP     Relocate
; Skip          .Byte   Version

ABuffer         .Equ    HostLocn+4
ACallSegTab     .Equ    ABuffer+2

B4ReadTest      .Equ    ACallSegTab+2
AfReadTest      .Equ    B4ReadTest+4
B4WriteTest     .Equ    AfReadTest+4
B4ElemsTest     .Equ    B4WriteTest+4

; Variables common to all routines
StrWork3        .Equ    B4ElemsTest+4
MainInMem       .Equ    StrWork3+4
TaskTable       .Equ    MainInMem+1
PrevSegLoad     .Equ    TaskTable+4

CallSegTab      .Equ    PrevSegLoad+1

CByte           .Equ    CallSegTab+PosSegsX4+4
CodePathName    .Equ    CByte+1

StrikeOver      .Equ    CodePathName+30.
CursorOnSw      .Equ    StrikeOver+1
LeaveAlone      .Equ    CursorOnSw+1
ChUnderCur      .Equ    LeaveAlone+2
SaveEI          .Equ    ChUnderCur+1

AskForProg      .Equ    SaveEI+1
CallSeg         .Equ    AskForProg+3
CC2S            .Equ    CallSeg+3
ClearDA         .Equ    CC2S+3
ClearTA         .Equ    ClearDA+3
ClearWindow     .Equ    ClearTA+3
ConsDriver      .Equ    ClearWindow+3
ConvITP         .Equ    ConsDriver+3
DieNow          .Equ    ConvITP+3
DivWord         .Equ    DieNow+3
DoBell          .Equ    DivWord+3
DoGoToXY        .Equ    DoBell+3
DoHelp          .Equ    DoGoToXY+3
DTTestAvail     .Equ    DoHelp+3
DTIsFull        .Equ    DTTestAvail+3
EnterCommand    .Equ    DTIsFull+3
FullExit        .Equ    EnterCommand+3  ; Z is to see if called anywhere
GetScrLine      .Equ    FullExit+3
GetArgs         .Equ    GetScrLine+3
GetVBar         .Equ    GetArgs+3
HighLight       .Equ    GetVBar+3
```

-- continued from previous page

```
IODisable       .Equ    HighLight+3
IOEnable        .Equ    IODisable+3
MakeIB          .Equ    IOEnable+3

MadeAChange     .Equ    MakeIB+3
MultByte        .Equ    MadeAChange+3
MultWord        .Equ    MultByte+3
MvLeftRtn       .Equ    MultWord+3
MvRightRtn      .Equ    MvLeftRtn+3
PopStack        .Equ    MvRightRtn+3
PressAny        .Equ    PopStack+3
Print           .Equ    PressAny+3
PushStack       .Equ    Print+3
ReadKB          .Equ    PushStack+3
RestCursor      .Equ    ReadKB+3
SaveGoToXY      .Equ    RestCursor+3
SetUCC          .Equ    SaveGoToXY+3
SetUCS          .Equ    SetUCC+3
StrCmpRtn       .Equ    SetUCS+3
StrMvRtn        .Equ    StrCmpRtn+3
StrWrRtn        .Equ    StrMvRtn+3
Trap            .Equ    StrWrRtn+3
VBarWrRtn       .Equ    Trap+3
Wait            .Equ    VBarWrRtn+3
WipeOutTA       .Equ    Wait+3
Write           .Equ    WipeOutTA+3
Writ1Byte       .Equ    Write+3
WritePE         .Equ    Writ1Byte+3
WritPRtn        .Equ    WritePE+3
WriteCom        .Equ    WritPRtn+3
WriteRUSure     .Equ    WriteCom+3

-- End of file.6
```

--- FILE.7 ---

Sub-host Documentation

```
; Book SubHost contains many of the host routines.

SMCountAvail

; SMCountAvail returns accum with count of K-bytes still available in the
; pool

SMGetCell

; SMGetCell returns Newstr with the contents of a specific cell of a specific
; record.  This routine should try to be fast.
;    1.  Address (bank pointer) for this record.
;    2.  In accum, category, or column, number.
; Returns Newstr with contents of area.
; Returns true or false whether Newstr has anything.

SMGetBlock

; SMGetBlock gets a block of information from memory pool.
;    1.  Address of a bank pointer.
;    2.  Where it goes in work area (Actual).
; CArg is returned with number of bytes gotten.

SMGetStrings

; SMGetStrings loads work area with string data, and CRSPT with the addresses
; of these various strings.  Accum has number of fields wanted.  If data has
; too many strings, ignore remainder.  If data has too few, fill the
; remaining pointers with zeroes.  Data ends with FF byte.
;    1.  Address of a bank pointer.
;    2.  Where the pointers go (Actual).
;    3.  Where the strings go (Actual).

SMPutBlock

; SMPutBlock puts a block into the memory pool.  If possible, uses previous
; address in pool.  Otherwise releases the previous address.
;    1.  Address of a bank pointer.
;    2.  Address of area to store (Actual).
;    3.  Length (bytes) of area to store: CArg.
; Returns:  New address in pool in whereever parm 1 points.
; Returns:  True or false, able to do it.
; This ain't gonna be fun on the //e.

SMPutCell

; SMPutCell replaces the contents of a specific cell.  This is used only by
; the spread sheet, during recalculation and adjust entries.
;    1.  Address (bank pointer) for this record.
;    2.  In accum, column, number.
;    3.  Newstr must have new data for cell.  It must be the same length as
;        the current contents of the cell.

SMPutStrings

; SMPutStrings puts string data into the memory pool.  # of fields in accum.
; If possible, uses previous address in pool.  Otherwise releases the
; previous address.
;    1.  Address of a bank pointer.
;    2.  Address of pointers (Actual).
; Returns:  New address in pool in whereever parm 1 points.
; Returns: True or false in accum.
; Fun, o fun, on the //e.
```

```
SMRelBlock

; SMRelBlock releases a block.
;    1.  Address of a bank pointer.
; Since the address is no longer valid, it is cleared.

SMReturnSize

; SMReturnSize returns CArg containing the number of bytes of real data for
; the specified area.
;    1.  Address of a bank pointer.

ReleaseKB

; ReleaseKB goes through the klipboard and frees up any memory that is in
; use.  Watch out for DO or more in second byte, means Typewriter formatting,
; not bank switched address.

ConvITDec

; ConvITDec converts accum to 'xx.x inches'  in Newstr.

DispEsc

; Routine to display Escape situation, and Help info.

DispFE

; Routine to display function (centered, top line of screen) and escape map.

DispFR

; Routine to display file name, if any, on top left line of screen.

DivWord

; Routine to Divide two words.  Addresses follow the JSR DivWord.
; Args: A (First)  divided by A(second), Result in MReg.

GetStr

; Routine to read a string.  Max Length is in Accum.
; Returns FoundEscape, FouncCommand, FoundReturn.

GetMenuBar

; This is quite a little routine to do inversed menu bars.  Let the user
; select an item by typing its NUMBER, or pressing return when the item is
; highlighted.  Use arrows to move highlighting.  MBP means Menu Bar
; processing.  Upon exit, capitalize the select one.
; Following comes from call
MBPEscape       .Word                   ; Where to go if escape hit
MBPCount        .Byte                   ; Number of params
MBPOptData      .Word                   ; Addr of description
MBPOptRtn       .Word                   ; Routine to process this opt.
                .Block          28.     ; 7 more options.
                .Block          5       ; Buffer space.

GetDec

; Routine to get a number between 0 and 25.6.  Pack into one byte (accum).

GetNum

; Routine to get a number.  Accum has maximum permissible value.  Returns in
; accum.  Zero means that escape was found.
```

```
GetYN

; Uses menu bar routines to get a yes or no.  Zero in accum means escape.

POS

; Routine to find position of first string in second string.  First string
; must be upper case, second can be upper/lower.  It will match upper against
; lower, etc. and still find equal.  Stolen from Leventhal Page 357.
; Improved as needed.
; Accum has (starting position-1)  in second string--usually zero.

STR

; Routine to convert a two-byte integer into printable.  Result in Newstr,
; left jusified.  Length of Newstr is from 1 to 5, depending on the number of
; significant digits.  Watch it, it is tricky and fast and tight.

UndoHelp

; Routine to undo the help on screen.  Puts K available on screen.

VerifyDelete

; VerifyDelete puts a box on the screen warning the user that something big
; is about to be killed.  Asks for a yes/no response.

-- End of file.7
```

--- FILE.8 ---

Sub-Host Entry Points

```
; These are references to the routines in SubHost
SMAvailCount    .Equ    LSubHost+2              ; 1.3 name changed

SMGetBlock      .Equ    SMAvailCount+3
SMGetCell       .Equ    SMGetBlock+3
SMGetStrings    .Equ    SMGetCell+3
SMMaxCol        .Equ    SMGetStrings+3
SMPutBlock      .Equ    SMMaxCol+3
SMPutCell       .Equ    SMPutBlock+3
SMPutStrings    .Equ    SMPutCell+3
SMRelBlock      .Equ    SMPutStrings+3
SMRetBSize      .Equ    SMRelBlock+3            ; 1.3 Name changed
SMRetWPSize     .Equ    SMRetBSize+3            ; 1.3 Name changed

Conv1TDec       .Equ    SMRetWPSize+3
DispEsc         .Equ    Conv1TDec+3
DispFR          .Equ    DispEsc+3
DispFE          .Equ    DispFR+3
GetDec          .Equ    DispFE+3
GetMenuBar      .Equ    GetDec+3
GetNum          .Equ    GetMenuBar+3
GetStr          .Equ    GetNum+3
GetYN           .Equ    Getstr+3
POSN            .Equ    GetYN+3
ReleaseKB       .Equ    POSN+3
Word2Str        .Equ    ReleaseKB+3             ; 1.3 name changed.
UndoHelp        .Equ    Word2Str+3
VerifyDelete    .Equ    UndoHelp+3
FirstEmpty      .Equ    VerifyDelete+3          ; Two bytes

-- End of file.8
```

--- FILE.4 ---

AppleWorks 2.0 Routine Hooks

This is documentation of hooks in 2.0 Aplworks.system. These hooks appear in the source code file "host1." Each is four bytes, and normally an RTS followed by three bytes of garbage. You can change these locations to JMP or JSR to your routines.

The computer will be in Apple //e emulation mode whenever the hooks are invoked. In some cases, the contents of the accumulator are important.

1. B4ReadTest: AppleWorks will JSR to this location before trying to read a character from the keyboard. The accumulator will be zeroed before the JSR. When AppleWorks regains control, the character in the accumulator, if not zero, will be processed as if it came from the keyboard.

   AppleWorks has a type-ahead buffer. Any characters in this buffer will be processed before B4ReadTest is attempted. You can place characters in the type-ahead buffer. It is documented below.

2. AfReadTest: AppleWorks will JSR to this location after it has read a character from any source. When AppleWorks regains control, the character in the accumulator, if not zero, will be processed normally. The $80 bit on each character is used to indicate that the Open Apple key was down.

   Watch out that characters you provided at B4ReadTest will be handed right back to you at AfReadTest.

   This feature added to version 2.0 causes the Control-@ character to be ignored, no matter how hard you press on the keys. So:

   WARNING: All versions of AppleWorks after, but not including, 2.0, will test the carry bit. If you want these versions to process the character in the accumulator, make sure that the carry bit is clear. The character in the accumulator will be ignored if you set the carry bit.

3. B4WriteTest: AppleWorks will JSR to this location before processing console driver commands. At entry the accumulator will contain a count of the number of bytes to be processed by the console driver. When AppleWorks regains control, the accumulator must contain the count, which you may change. If you change the count to zero, the console driver will do nothing.

   Zero page Word CDAddr has the address of the first byte to be processed by the console driver. You shouldn't change the characters to be processed, but you can change the zero page word to point at different characters in your work area.

   Various hex values less than $20 have special meaning to the console driver, and are listed in the file iall/common as ClearLine through HorShiftVP. There are many similarities to the Apple /// ".console" driver.

4. B4ElemsTest: AppleWorks will JSR to this location before making calls to the SANE elementary functions package.

5. The type-ahead buffer is defined in all/common. It is 24 bytes, and you can add to it at any time.

   TANextIn and TANextOut have values from 0 to 23. If they are equal, then the type-ahead is empty. TANextIn is an index to the next position available to store another character in TABuffer. TANextOut is an index to the next character to be provided from TABuffer.

-- End of file.4

Desktop Manager Entry Points (2.1 LAN code)

```
; File of references to Organizer
;               Equates for all segments
DDR             .Equ            FBR             ; ODO
IOMustDo        .Equ            ZP+0D4          ; Boolean
RWRefNo         .Equ            ZP+0D5
LDDEntry        .Equ            24.
DDDBType        .Equ            19
DDWPType        .Equ            1A
DDSSType        .Equ            1B

AskForData      .Equ            LOrgMain+2+3
AskPathName     .Equ            AskForData+3
BuildPrefix     .Equ            AskPathName+3
ChangeFC        .Equ            BuildPrefix+3
CloseAFile      .Equ            ChangeFC+3
ConvH2PC        .Equ            CloseAFile+3
Display2SOS     .Equ            ConvH2PC+3
DrawFileCard    .Equ            Display2SOS+3
FCList          .Equ            DrawFileCard+3
Hitting12       .Equ            FCList+3        ; NOTE THREE BYTES AVAILABLE.
LoadDied        .Equ            Hitting12+3
MsgMaxRecs      .Equ            LoadDied+3
OpenInput       .Equ            MsgMaxRecs+3
ProvName        .Equ            OpenInput+3
ReadVar         .Equ            ProvName+3
SlotClrCurr     .Equ            ReadVar+3
SlotFrCurr      .Equ            SlotClrCurr+3
Slot2Curr       .Equ            SlotFrCurr+3
SlotFrWork      .Equ            Slot2Curr+3
Slot2Work       .Equ            SlotFrWork+3
SlotPoint       .Equ            Slot2Work+3
SOSJustClose    .Equ            SlotPoint+3
SOS2Display     .Equ            SOSJustClose+3
WrapFileName    .Equ            SOS2Display+3
WrRel2Card      .Equ            WrapFileName+3

; First the main entry  ; Reg X has call parameter
AFileBuf        .Equ            WrRel2Card+3

; Note 36 bytes available after Prefix.
Prefix          .Equ            AFileBuf+2      ; 50. bytes
RWFirst         .Equ            Prefix+86.      ; To maintain location
EndFileBuf      .Equ            RWFirst+1



DDBot           .Equ            EndFileBuf+2
DDTop           .Equ            DDBot+2
DDTopT          .Equ            DDTop+2
```

-- continued next page --

-- continued previous page --

```
; Work area for one entry from my internal disk directory
DDName          .Equ            DDTopT+2
DDType          .Equ            DDName+16.
DDSizeK         .Equ            DDType+1        ; 1.3 changed to 2 bytes
DDDate          .Equ            DDSizeK+2
DDPicked        .Equ            DDDate+4
;                                               ; 1.3 One spare byte removed

FCLLoaded       .EQU            DDName+LDDEntry ; 1.3 This doesn't change
locn.
FCLPicked       .EQU            FCLLoaded+1
APickMsg        .Equ            FCLPicked+1     ; Address of "Use arrows 2
pick"
QDSaveSw        .Equ            APickMsg+2      ; Doing Quick'n'Dirty Save

;               Work area for loading files from BS
WDTFName        .Equ            QDSaveSw+1
WDTFType        .Equ            WDTFName+21.
WDTFStatus      .Equ            WDTFType+1
WDTFPicked      .Equ            WDTFStatus+1    ; Whether picked for save
WDTFSizK        .Equ            WDTFPicked+1    ; 1.3 two bytes
WDTFAddress     .Equ            WDTFSizK+2      ; Bank address if not curr.

WorkFileName    .Equ            WDTFName+32.    ; 66. bytes

-- end of file.27
```

Data Base Common Work Area

```
; Equates and zero page locations for Data base

Task            .Equ            "F"             ; ie., Quick File
MainSegNo       .Equ            1               ; Main is seg.01
MaxFields       .Equ            30.
MaxRpWidth      .Equ            180.
RptSizeH        .Equ            620.            ; Bytes for tables style
RptSizeV        .Equ            450.            ; For labels style
VRLine1         .Equ            7
KAuxBase        .Equ            30              ; Rel addresses >= are in aux.

; For report formats (Tables style)
LRptArea        .Equ            600.
LCalcArea       .Equ            54.

; Zero page addresses

BR              .Equ    ZP+0A8                  ; For file loading
CR              .Equ    ZP+0AA                  ; Column register
HR              .Equ    ZP+0AE                  ; Just in case
RR              .Equ    ZP+0B0

                .If             Language="A"
ComArrange      .Equ                "A"+0AKey
ComCopy         .Equ                "C"+0AKey
ComDitto        .Equ                "'"+0AKey
ComDelete       .Equ                "D"+0AKey
ComFind         .Equ                "F"+0AKey
ComGroup        .Equ                "G"+0AKey
ComInsert       .Equ                "I"+0AKey
ComJustify      .Equ                "J"+0AKey
ComCalcCol      .Equ                "K"+0AKey
ComLayout       .Equ                "L"+0AKey
ComMove         .Equ                "M"+0AKey
ComChName       .Equ                "N"+0AKey
ComPrOptions    .Equ                "O"+0AKey
ComPrint        .Equ                "P"+0AKey
ComRecSel       .Equ                "R"+0AKey
ComTotals       .Equ                "T"+0AKey
ComStandard     .Equ                "V"+0AKey
ComPrFN         .Equ                "Y"+0AKey
ComZoom         .Equ                "Z"+0AKey
ComPAGEBACK     .Equ                11.+0AKey
ComPAGEFORW     .Equ                10.+0AKey
                .Endc

                .If             Language="F"
ComArrange      .Equ                "T"+0AKey
ComCopy         .Equ                "C"+0AKey
ComDitto        .Equ                22+0AKey
ComDelete       .Equ                "E"+0AKey
ComFind         .Equ                "L"+0AKey
ComGroup        .Equ                "G"+0AKey
ComInsert       .Equ                "I"+0AKey
ComJustify      .Equ                "J"+0AKey
ComCalcCol      .Equ                "K"+0AKey
ComLayout       .Equ                "X"+0AKey
ComMove         .Equ                "D"+0AKey
ComChName       .Equ                "N"+0AKey
ComPrOptions    .Equ                "O"+0AKey
ComPrint        .Equ                "P"+0AKey
ComRecSel       .Equ                "R"+0AKey
ComTotals       .Equ                "M"+0AKey
ComStandard     .Equ                "V"+0AKey
ComPrFN         .Equ                "V"+0AKey
ComZoom         .Equ                "Z"+0AKey
ComPAGEBACK     .Equ                11.+0AKey
ComPAGEFORW     .Equ                10.+0AKey
                .Endc
```

```
                .If             Language="G"
ComArrange      .Equ            "O"+0AKey
ComCopy         .Equ            "K"+0AKey
ComDitto        .Equ            22+0AKey
ComDelete       .Equ            "L"+0AKey
ComFind         .Equ            "F"+0AKey
ComGroup        .Equ            "G"+0AKey
ComInsert       .Equ            "E"+0AKey
ComJustify      .Equ            "J"+0AKey
ComCalcCol      .Equ            "K"+0AKey
ComLayout       .Equ            "L"+0AKey
ComMove         .Equ            "B"+0AKey
ComChName       .Equ            "N"+0AKey
ComPrOptions    .Equ            "P"+0AKey
ComPrint        .Equ            "D"+0AKey
ComRecSel       .Equ            "R"+0AKey
ComTotals       .Equ            "T"+0AKey
ComStandard     .Equ            "V"+0AKey
ComPrFN         .Equ            "V"+0AKey
ComZoom         .Equ            "Z"+0AKey
ComPAGEBACK     .Equ            11.+0AKey
ComPAGEFORW     .Equ            10.+0AKey
                .Endc

                .If             Language="I"
ComArrange      .Equ            "O"+0AKey
ComCopy         .Equ            "D"+0AKey
ComDelete       .Equ            "E"+0AKey
ComDitto        .Equ            22+0AKey
ComFind         .Equ            "C"+0AKey
ComGroup        .Equ            "G"+0AKey
ComInsert       .Equ            "I"+0AKey
ComJustify      .Equ            "J"+0AKey
ComCalcCol      .Equ            "K"+0AKey
ComLayout       .Equ            "L"+0AKey
ComMove         .Equ            "M"+0AKey
ComChName       .Equ            "N"+0AKey
ComPrOptions    .Equ            "P"+0AKey
ComPrint        .Equ            "S"+0AKey
ComRecSel       .Equ            "R"+0AKey
ComTotals       .Equ            "T"+0AKey
ComStandard     .Equ            "V"+0AKey
ComPrFN         .Equ            "V"+0AKey
ComZoom         .Equ            "Z"+0AKey
ComPAGEBACK     .Equ            139.
ComPAGEFORW     .Equ            138.
                .Endc

; The following is the layout of the Quick File work area.  The beginning
; address is somewhat variable until this settles down.
                .List

;               First the items that are common between all files

; Begin the file header record
DBHeader        .Equ            DBWA
A8Reports       .Equ            DBHeader        ; Address of record pointers
BABARecords     .Equ            A8Reports+16.   ; Bank adr of block of addr.
                                                ; of records

;               Now file records one and two, exactly

BasicMode       .Equ            DBWA+24.
CurRecNo        .Equ            BasicMode+1
CurrRpNo        .Equ            CurRecNo+2
FVSequence      .Equ            CurrRpNo+1
FHRetDir        .Equ            FVSequence+1
HDCLine         .Equ            FHRetDir+1
HDCColumn       .Equ            HDCLine+1
Mode            .Equ            HDCColumn+1
NFields         .Equ            Mode+1
NRecords        .Equ            NFields+1
NReports        .Equ            NRecords+2
PSL             .Equ            NReports+1
VDCField        .Equ            PSL+2
```

```
;                       Blocks of info for up to 30 fields, plus some slack

FHHor                   .Equ            VDCField+1
FHField                 .Equ            FHHor+36.
FVHor                   .Equ            FHField+36.
FVVer                   .Equ            FVHor+36.
FVField                 .Equ            FVVer+36.
FHFields                .Equ            FVField+36.

;                       Select data from file.  Has to stay as shown in Pascal

SelectOn                .Equ            FHFields+1
SelectTest              .Equ            SelectOn+6
SelectCont              .Equ            SelectTest+6
SelectData              .Equ            SelectCont+6
Filler1                 .Equ            SelectData+96.
FHNames                 .Equ            Filler1+20.

;                       This is the end of the exact format of the input file

; The number of FHNames stored will be same as NFields; 22 bytes each.

;                       Work area to store record pointers.  1025 for //e,
;                       3073 for ///.  Record 0 is defaults, 1-n are real.

BldRecWA                .Equ            FHNames+660.
BldRecTop               .Equ            BldRecWA+1024.

; Information that is built while a file is open, and not important
; to save when closed


CRSPT                   .Equ            BldRecTop
CurrRpName              .Equ            CRSPT+60.
ConvError               .Equ            CurrRpName+21.
ConvMore                .Equ            ConvError+1
DateIn                  .Equ            ConvMore+1
DateGood                .Equ            DateIn+21.
XFILLER                 .Equ            DateGood+1
FFSWitch                .Equ            XFILLER+1
FFData                  .Equ            FFSWitch+1
IsDateF                 .Equ            FFData+32.
IsTimeF                 .Equ            IsDateF+1
L15Work                 .Equ            IsTimeF+1
MaxLength               .Equ            L15Work+16.
Numstr                  .Equ            MaxLength+1
NFieldsX2               .Equ            NumStr+6
NewStr1                 .Equ            NFieldsX2+1
SaveCWork1              .Equ            Newstr1+1
SFDisplace              .Equ            SaveCWork1+1
SortL                   .Equ            SFDisplace+1
SaveFR                  .Equ            SortL+1
TopRecNo                .Equ            SaveFR+2
WorkPointer             .Equ            TopRecNo+2

; Following are not kept on file
SelectType              .Equ            WorkPointer+2
SelectNumeric           .Equ            SelectType+6

; Save selection rules while in del mode, or report
SaveRecSel              .Equ            SelectNumeric+48.
IFCEnd                  .Equ            SaveRecSel+114.

; Area is 1350*2 or 7350*2, depending on whether Slinky present.
ARecords                .Equ            IFCEnd

-- End of file.1
```

Data Base Common Work Area, Additional

```
; THE FOLLOWING MUST BE IN ORDER, AS THEY COME FROM DISK *
RptName         .Equ            RWFormat
RHor            .Equ            RptName+20.     ; Horizontal
RVer            .Equ            RHor+36.        ; Vertical
RField          .Equ            RVer+36.        ; Field number for column
                                                ; 80-82 are calced.
RFoot           .Equ            RField+36.      ; H: $99 no foot total,
                                                ;    0-4 is decimal places
                                                ; V: Boolean print field name
RFloat          .Equ            RFoot+36.       ; H: $99 not numeric field
                                                ;    0-4 decimal places
                                                ; V: Boolean float left
RFields         .Equ            RFloat+36.      ; Fields on reports
RCCol           .Equ            RFields+1       ; H: 1st Calc col of RField array
                                                ; V: Screen line bot of page
                                                ; Then two more for 2nd,3rd calc.
RSTCol          .Equ            RCCol+3         ; Of FHNames array

RWidth          .Equ            RSTCol+1        ; Of paper inches XX.X
RLMargin        .Equ            RWidth+1        ; Inches XX.X
RRMargin        .Equ            RLMargin+1      ; Inches XX.X
RChPerInch      .Equ            RRMargin+1      ; NN

RLPaper         .Equ            RChPerInch+1    ; Inches XX.X
RTMargin        .Equ            RLPaper+1       ; Inches XX.X
RBMargin        .Equ            RTMargin+1      ; Inches XX.X
RLinesPerInch   .Equ            RBMargin+1

RSource         .Equ            RLinesPerInch+1 ; Char New Changed source
RType           .Equ            RSource+1       ; Char H or Vertical format
RSpacing        .Equ            RType+1         ; Char S D or T, all languages
RPrTitle        .Equ            RSpacing+1      ; Boolean print title block
RJustTot        .Equ            RPrTitle+1      ; Boolean Just print totals
ROmitBlank      .Equ            RJustTot+1      ; Boolean
RPrIBlank       .Equ            ROmitBlank+1    ; Boolean
RTitle          .Equ            RPrIBlank+1     ; Title line
RCalcName       .Equ            RTitle+82.      ; Max 20, calc name (3 of them)
RCalcRule       .Equ            RCalcName+22.   ; Max 30, rules (3 of them)
RC2and3         .Equ            RCalcRule+32.   ; Then two more, 54 bytes each.
RSTPage         .Equ            RC2and3+LCalcArea+LCalcArea
RPCC            .Equ            RSTPage+1       ; Max 13 control char for pr.
RDashes         .Equ            RPCC+14.        ; Boolean print dashes for empty

RFSelRules      .Equ            RDashes+1       ; 114 bytes
```

-- end of file.35

--- FILE.32 ---

Data Base Entry Points

```
; These are the entry points into Main

AddNum2S          .Equ    LDBMain+2+3
ClearCC           .Equ    AddNum2S+3
ClearFileHeader   .Equ    ClearCC+3
ConvS2I14         .Equ    ClearFileHeader+3
CCSelect          .Equ    ConvS2I14+3
DecDate           .Equ    CCSelect+3
DecTime           .Equ    DecDate+3
DispCC            .Equ    DecTime+3
DispTime          .Equ    DispCC+3
DispDate          .Equ    DispTime+3
DispRN            .Equ    DispDate+3
DumpCRSPT         .Equ    DispRN+3
ExtractN          .Equ    DumpCRSPT+3
FormICC           .Equ    ExtractN+3
LoadCRSPT         .Equ    FormICC+3
PointAtNames      .Equ    LoadCRSPT+3
PointAtRec        .Equ    PointAtNames+3
ResetRAC          .Equ    PointAtRec+3
SetCCType         .Equ    ResetRAC+3
StuffName         .Equ    SetCCType+3
TryDateTime       .Equ    StuffName+3
TestDtTime        .Equ    TryDateTime+3


; Now the data elements that are loaded as part of Main
DashStr           .Equ            TestDtTime+3
S1                .Equ            DashStr+3
ALONames          .Equ            S1+2

ALCNames          .Equ            ALONames+36.      ; 18*2
MonthNames        .Equ            ALCNames+6
```

-- end of file.32

--- FILE.33 ---

Data Base Report Work Area

```
;                Zero page locations for RWSeg
LeftMargin       .Equ     063
RightMargin      .Equ     064
CurrColumn       .Equ     065
CurrHor          .Equ     066
CurrVer          .Equ     067
LeftColumn       .Equ     068
VRCurrField      .Equ     069
CurHorFull       .Equ     06A
CurHorScr        .Equ     06B
NxtHorFull       .Equ     06C
MenuItem         .Equ     06D
MadeRPChanges    .Equ     06E
FirstRC          .Equ     06F
OPAscii          .Equ     070
Op2Scr           .Equ     071
Op2HC            .Equ     072
Op2Disk          .Equ     073
Op2WP            .Equ     074
Op2MM            .Equ     075
OPDIF            .Equ     076

; Location of report format
RWFormat         .Equ           LDBSegs


; ALL THE FOLLOWING TABLES MUST BE IN SOME SORT OF ORDER, AS THEY ARE BULK
; MOVED INTO THE AREAS.  Max 512 bytes.

; Table of spacing intervals for various printers.
; Values are for 9, thru 17 cpi.  All values are based on non-proportional
; letters taking 8 increments.  Values 1 and 2 are for proportional 1 and 2.
; These are default values.  Can be overridden for specific printers.


PrAreaStart      .Equ           RWFormat+LRptArea  ; Dots per inch for 0-24 CPI

-- end of file.33
```

--- FILE.34 ---

Data Base Report Work Area Entry Points

```
; Book RWSegRef
; Routines

DispTop          .Equ            LDBSegs+0500+2+3
HRDispL5         .Equ            DispTop+3
Inch2Chars       .Equ            HRDispL5+3
Inch2Lines       .Equ            Inch2Chars+3


SelPrType        .Equ            Inch2Liness+3
SelOptions       .Equ            SelPrType+1
```

-- end of file.34

Spreadsheet Common Work Area

```
; Common routines for SpreadSheet assembly
;
MainSegNo       .Equ    24.             ; Main for SS is seg.24
Task            .Equ    "C"

                .If     Language="A"
ComArrange      .Equ    "A"+0AKey
ComBlank        .Equ    "B"+0AKey
ComCopy         .Equ    "C"+0AKey
ComDelete       .Equ    "D"+0AKey
ComFind         .Equ    "F"+0AKey
ComInsert       .Equ    "I"+0AKey
ComJump         .Equ    "J"+0AKey
ComCalc         .Equ    "K"+0AKey
ComLayout       .Equ    "L"+0AKey
ComMove         .Equ    "M"+0AKey
ComChName       .Equ    "N"+0AKey
ComOptions      .Equ    "O"+0AKey
ComPrint        .Equ    "P"+0AKey
ComTitles       .Equ    "T"+0AKey
ComEdit         .Equ    "U"+0AKey
ComStandard     .Equ    "V"+0AKey
ComView         .Equ    "W"+0AKey
ComZoom         .Equ    "Z"+0AKey
KFormSep        .Equ    ","             ; Separates list items in formula
                .Endc


                .If     Language="F"
ComArrange      .Equ    "T"+0AKey
ComBlank        .Equ    "B"+0AKey
ComCopy         .Equ    "C"+0AKey
ComDelete       .Equ    "E"+0AKey
ComFind         .Equ    "L"+0AKey
ComInsert       .Equ    "I"+0AKey
ComJump         .Equ    "J"+0AKey
ComCalc         .Equ    "K"+0AKey
ComLayout       .Equ    "X"+0AKey
ComMove         .Equ    "D"+0AKey
ComChName       .Equ    "N"+0AKey
ComOptions      .Equ    "O"+0AKey
ComPrint        .Equ    "P"+0AKey
ComTitles       .Equ    "M"+0AKey
ComEdit         .Equ    "U"+0AKey
ComStandard     .Equ    "V"+0AKey
ComView         .Equ    "F"+0AKey
ComZoom         .Equ    "Z"+0AKey
KFormSep        .Equ    ";"             ; Separates list items in formula
                .Endc

                .If     Language="G"
ComArrange      .Equ    "O"+0AKey
ComBlank        .Equ    "R"+0AKey
ComCopy         .Equ    "K"+0AKey
ComDelete       .Equ    "L"+0AKey
ComFind         .Equ    "F"+0AKey
ComInsert       .Equ    "E"+0AKey
ComJump         .Equ    "Z"+0AKey
ComCalc         .Equ    "M"+0AKey
ComLayout       .Equ    "I"+0AKey
ComMove         .Equ    "B"+0AKey
ComChName       .Equ    "N"+0AKey
ComOptions      .Equ    "P"+0AKey
ComPrint        .Equ    "D"+0AKey
ComTitles       .Equ    "H"+0AKey
ComEdit         .Equ    "V"+0AKey
ComStandard     .Equ    "G"+0AKey
ComView         .Equ    "T"+0AKey
ComZoom         .Equ    "A"+0AKey
KFormSep        .Equ    ";"             ; Separates list items in formula
                .Endc
```

```
                    .If     Language="I"
ComArrange          .Equ    "O"+0AKey
ComBlank            .Equ    "B"+0AKey
ComCopy             .Equ    "D"+0AKey
ComDelete           .Equ    "E"+0AKey
ComFind             .Equ    "C"+0AKey
ComInsert           .Equ    "I"+0AKey
ComJump             .Equ    "J"+0AKey
ComCalc             .Equ    "K"+0AKey
ComLayout           .Equ    "L"+0AKey
ComMove             .Equ    "M"+0AKey
ComChName           .Equ    "N"+0AKey
ComOptions          .Equ    "P"+0AKey
ComPrint            .Equ    "S"+0AKey
ComTitles           .Equ    "T"+0AKey
ComEdit             .Equ    "U"+0AKey
ComStandard         .Equ    "V"+0AKey
ComView             .Equ    "F"+0AKey
ComZoom             .Equ    "Z"+0AKey
KFormSep            .Equ    ";"                 ; Separates list items in formula
                    .Endc

; Bits for WorkType byte +0
BitValue            .Equ    80
BitNDIfZero         .Equ    40              ; Values solamente
BitProp             .Equ    20              ; Just for labels
BitSimValue         .Equ    20              ; Just for values.
BitProNL            .Equ    10              ; Labels not allowed
BitProNV            .Equ    8               ; Values not allowed

BitsVFS             .Equ    1               ; Bits Value Format Standard
BitsVFF             .Equ    2               ; For values
BitsVFD             .Equ    3
BitsVFC             .Equ    4
BitsVFP             .Equ    5
BitsVFA             .Equ    6

BitsLFS             .Equ    1
BitsLFL             .Equ    2               ; For labels
BitsLFR             .Equ    3
BitsLFC             .Equ    4

; Bits for WorkType byte +1 (just for values)
BitCalced           .Equ    80
BitNA               .Equ    40
BitError            .Equ    20
BitChanged          .Equ    10              ; This is probably not useable

; These are internal codes found in internally stored strings
KIntCoor            .Equ                254.        ; Next two bytes are XY
KIntNum             .Equ                253.        ; Next six bytes are number
KIntDots            .Equ                252.        ; Replaces ...
KAUnPlus            .Equ                251.
KAUnMinus           .Equ                250.
KALParen            .Equ                249.
KAMult              .Equ                248.
KADiv               .Equ                247.
KAPlus              .Equ                246.
KAMinus             .Equ                245.
KARParen            .Equ                244.
KAExp               .Equ                243.
KAComma             .Equ                242.        ; Comma within function

;                   Logical operators
KALessThan          .Equ                241.
KAGtrThan           .Equ                240.
KAEquals            .Equ                239.
KALessEqual         .Equ                238.
KAGtrEqual          .Equ                237.
KANotEqual          .Equ                236.
```

— continued next page —

-- continued previous page --

```
;                This is just used by replicate
KAbsCoor         .Equ            210.

;                Now start the functions
KFRound          .Equ            217.
KFOr             .Equ            218.
KFAnd            .Equ            219.
KFSum            .Equ            220.    ; First because most common
KFAvg            .Equ            221.
KFChoose         .Equ            222.
KFCount          .Equ            223.
KFError          .Equ            224.
KFIRR            .Equ            225.
KFIF             .Equ            226.
KFINT            .Equ            227.
KFLookup         .Equ            228.
KFMax            .Equ            229.
KFMin            .Equ            230.
KFNA             .Equ            231.
KFNPV            .Equ            232.
KFSQRT           .Equ            233.
KFAbs            .Equ            234.

;                Now some constants
KHighCol         .Equ            127.
KFirstDLine      .Equ            2
KDispLines       .Equ            18.     ; 18 lines on screen
KLastDLine       .Equ            19.
LIntNum          .Equ            8       ; Length of internal numerics storage

; Zero page addresses for Quick Calc
CurRow           .Equ    ZP+0AB          ; Current Row (1-999)
CurCol           .Equ    ZP+0AD          ; Current Col (1-127)
WorkRow          .Equ    ZP+0AE          ; Used by many routines
WorkCol          .Equ    ZP+0B0          ; Used by many routines

; Provided by GetSpecs
CCWidth          .Equ    ZP+0B1          ; Width of current column
HOnScreen        .Equ    ZP+0B2          ; 0: no 1: partial, 2: fully
VOnScreen        .Equ    ZP+0B3          ; 0: no, 1: Yes
MaxLength        .Equ    ZP+0B4

; Provided by AccumType
IsParen          .Equ    ZP+0B5
IsOpr            .Equ    ZP+0B6
IsAlpha          .Equ    ZP+0B7
IsNum            .Equ    ZP+0B8
IsBuiltIn        .Equ    ZP+0B9

SwDispBox        .Equ    ZP+0BA      ; Redisplay the entire page, including MakeIB
SwDispI2         .Equ    ZP+0BB      ; Redisplay all rows within box
SwDispPl         .Equ    ZP+0BC      ; Redisplay line 23
SwDispCur        .Equ    ZP+0BD      ; Inverse the cursor
```

-- continued next page

-- continued previous page

```
ValidMoves      .Equ    ZP+0BE      ; Valid cursor moves, see next two lines
ValidRows       .Equ    80
ValidCols       .Equ    40
ValidAny        .Equ    80+40
DispFRows       .Equ    20      ; Tell expand cursor to display full rows
DispFCols       .Equ    10      ; Tell expand cursor to display full cols
ValidDim        .Equ    08      ; Only allow expand in one direction

IsArrow         .Equ    ZP+0BF
SwRecalc        .Equ    ZP+0C0

; Zero page locations for convert to internal
CX2IVal1        .Equ            ZP+0C1
CX2IVal2        .Equ            ZP+0C2

; Bits for the above two bytes, indicating what's valid right now
OKUnPM          .Equ            80      ; Unury plus or minus
OKOpr           .Equ            40      ; * + - / or ^
OKDots          .Equ            20      ; ... (between operators
OKRParen        .Equ            10      ; Right paren
OKAt            .Equ            08      ; @ sign
OKDecPt         .Equ            04      ; Begins a number
OKLParen        .Equ            02      ; Left paren
OKAlpha         .Equ            01      ; A to Z, starting coordinates
OKNum           .Equ            80      ; Start CX2IVal2, numeric
OKArrow         .Equ            40      ; Can use arrows to find cell
OKComma         .Equ            20      ; Comma is OK
OKEnd           .Equ            10      ; Ok to end now, by CR or arrow.


; Zero page assignment:
;  0AB - 0C8   Used throughout, defined here
;  0C9 - 0D8   Defined and used in Main only
;  0D9 - 0E4   Defined and used by EditM
;  0E4 - 0EF   Defined and used by EditE, EditL, etc.


; The following is the layout of the Quick Calc work area.  The beginning
; address is somewhat variable until this settles down.


; Information the file that must be kept while the file is in memory, but
; closed.
ColWidths       .Equ            SSWA+3          ; 128 bytes

;                Global variables
GloOrder        .Equ            ColWidths+128.  ; Initial: Appropriate
GloReCalc       .Equ            GloOrder+1      ; Initial: Automatic
SvLocn          .Equ            GloReCalc+1     ; CurRow,CurCol

; Information that controls the views
VIViews         .Equ    SvLocn+3        ; Split "1 S(ide by side, T(op and bottom
VISync          .Equ    VIViews+1       ; Boolean, sync or not if two views
```

-- continued next page

-- continued previous page

```
; Information for the current view, or window
VCBlock         .Equ    VISync+1

VCLFormat       .Equ    VCBlock         ; Standard for view--Initial: Left
VCVFormat       .Equ    VCLFormat+1     ; Standard:  Initial: Appro., 0 length
VCTLVer         .Equ    VCVFormat+2     ; Top left corner, ver on screen
VCTLHor         .Equ    VCTLVer+1       ; Top left corner, hor on screen

VCTTLRow        .Equ    VCTLHor+1       ; Titles, top left row (may be zero)
VCTTLCol        .Equ    VCTTLRow+2      ; Titles, top left col
VCTLastRow      .Equ    VCTTLCol+1
VCTLastCol      .Equ    VCTLastRow+2

VCBTLRow        .Equ    VCTLastCol+1    ; Body, top left row
VCBTLCol        .Equ    VCBTLRow+2      ; Body, top left col
VCBTLVer        .Equ    VCBTLCol+1      ; Body, top left ver (of first row)
VCBTLHor        .Equ    VCBTLVer+1      ; Body, top left hor (of first col)

VCBBRRow        .Equ    VCBTLHor+1      ; Body, bottom right row
VCBBRCol        .Equ    VCBBRRow+2      ; Body, bottom right col
VCBBRVer        .Equ    VCBBRCol+1      ; Ver coord of bottom right corner of box
VCBBRHor        .Equ    VCBBRVer+1      ; Hor coord.
VCBWidth        .Equ    VCBBRHor+1      ; Actual screen cols needed for
                                        ; displayed data
VCRPartial      .Equ    VCBWidth+1      ; Boolean--right col is partial
VCTitles        .Equ    VCRPartial+1    ; TOP: x'80', left side: x'40'

VCBlockE        .Equ    VCTitles+1
; End of View (current) parameters

; Information for the secondary view (ie where the cursor isn't).
VSBlock         .Equ    VCBlockE

VSLFormat       .Equ    VSBlock         ; Initial: Left
VSVFormat       .Equ    VSLFormat+1     ; Initial: Appro., 0 length
VSTLVer         .Equ    VSVFormat+2     ; Top left corner, ver on screen
VSTLHor         .Equ    VSTLVer+1

VSTTLRow        .Equ    VSTLHor+1
VSTTLCol        .Equ    VSTTLRow+2
VSTLastRow      .Equ    VSTTLCol+1
VSTLastCol      .Equ    VSTLastRow+2

VSBTLRow        .Equ    VSTLastCol+1
VSBTLCol        .Equ    VSBTLRow+2
VSBTLVer        .Equ    VSBTLCol+1
VSBTLHor        .Equ    VSBTLVer+1

VSBBRRow        .Equ    VSBTLHor+1
VSBBRCol        .Equ    VSBBRRow+2
VSBBRVer        .Equ    VSBBRCol+1
VSBBRHor        .Equ    VSBBRVer+1
VSBWidth        .Equ    VSBBRHor+1
VSRPartial      .Equ    VSBWidth+1
VSTitles        .Equ    VSRPartial+1

VSBlockE        .Equ    VSTitles+1
; End of View (secondary) parameters

; Area to save the current window so that exact position can be returned to
; during replicate and formula expansion.
SvVCBlock       .Equ    VSBlockE
SvVCCol         .Equ    VCBlockE-VCBlock+SvVcBlock
SvVCRow         .Equ    SvVcCol+1


;               More variables
Protection      .Equ    SvVCRow+2       ; Boolean, on or off
CameFrVC        .Equ    Protection+1    ; Boolean, if values need to be calced.
```

```
;               The following are printer formatting variables.  They
;               get initialized to zero with a new file.  The print
;               routines will initialize to real initial values.
RWidth          .Equ            CameFrVC+1      ; Of paper inches XX.X
RLMargin        .Equ            RWidth+1        ; Inches XX.X
RRMargin        .Equ            RLMargin+1      ; Inches XX.X
RChPerInch      .Equ            RRMargin+1      ; NN

RLPaper         .Equ            RChPerInch+1    ; Inches XX.X
RTMargin        .Equ            RLPaper+1       ; Inches XX.X
RBMargin        .Equ            RTMargin+1      ; Inches XX.X
RLinesPerInch   .Equ            RBMargin+1

RSpacing        .Equ            RLinesPerInch+1 ; Char S D or T, all languages
RPCC            .Equ            RSpacing+1      ; Max 13 control char for pr.
RDashes         .Equ            RPCC+14.        ; Boolean print dashes for empty
RPrTitle        .Equ            RDashes+1       ; Print title block on report

;               Keep track of zoomed in or not
ZoomInSw        .Equ            RPrTitle+1

;               Work area to store record pointers.  999 for either machine
ARecords        .Equ            SSWA+300.

BotRtRow        .Equ            ARecords+MaxQCRows+MaxQCRows+2
BotRtCol        .Equ            BotRtRow+2

ConvError       .Equ            BotRtCol+1
ConvMore        .Equ            ConvError+1
CRSPT           .Equ            ConvMore+1
CXISaveX        .Equ            CRSPT+256.
CXISWork        .Equ            CXISaveX+1
LJColNo         .Equ            CXISWork+128.
LJRowNo         .Equ            LJColNo+3.
LJCellID        .Equ            LJRowNo+4
LocFormat       .Equ            LJCellID+6

WorkType        .Equ            LocFormat+1     ; two bytes.  Values and
                                                ; propd labels use 1, reg
                                                ; labels use 1

WorkFPD         .Equ            WorkType+2      ; Values only
WorkVal         .Equ            WorkFPD+8       ; Values and non-prop labels

CX2IResult      .Equ            WorkVal+128.
CX2IRelative    .Equ            CX2IResult+1
PrevDAddr       .Equ            CX2IRelative+1

;               Stuff for highlighted box
BlAnchCol       .Equ            PrevDAddr+2     ; Cursor posn at entry
BlAnchRow       .Equ            BlAnchCol+1

;               SetCorners places the top left, and bottom right vvalues
BlSoStCol       .Equ            BlAnchRow+2     ; Top left corner
BlSoStRow       .Equ            BlSoStCol+1
BlSoOppCol      .Equ            BlSoStRow+2     ; Bot right corner
BlSoOppRow      .Equ            BlSoOppCol+1
```

```
;                   Destination row and column
BlDsStCol           .Equ            BlSoOppRow+2
BlDsStRow           .Equ            BlDsStCol+1
BlDsOppCol          .Equ            BlDsStRow+2     ; Bot right corner
BlDsOppRow          .Equ            BlDsOppCol+1

;                   Note that source and destination are switched during
;                   parts of Replicate.

ReplWork            .Equ            BlDsOppRow+2    ; 128. bytes
FFData              .Equ            ReplWork+128.   ; 30 bytes
FFType              .Equ            FFData+30.
FFCoor              .Equ            FFType+1
SSCaller            .Equ            FFCoor+6        ; Command that called segment
;                   CXIPType is
;                   0:  Left paren was not following an @ function
;                   non zero:  Is the parameters byte from the @functin table
CXIPType            .Equ            SSCaller+1      ; 20. bytes, just for EditM

; This next work area is for reading and writing whole rows.  The
; size is 1550 bytes on 128K system, 5000 bytes on Slinky systems.
BldRecWA            .Equ            CXIPType+24.

-- end of file.9
```

Spreadsheet Main Segment Entry Points

```
AccumType         .Equ          LSSMain+2+3
BldCellID         .Equ          AccumType+3
CellFrBS          .Equ          BldCellID+3
CellFrWA          .Equ          CellFrBS+3
CellToBS          .Equ          CellFrWA+3
Cell2WA           .Equ          CellToBS+3        ; Need to load VC file.
ClearCRSPT        .Equ          Cell2WA+3
ClearFileHeader   .Equ          ClearCRSPT+3
ClearL22          .Equ          ClearFileHeader+3
ConvertColNo      .Equ          ClearL22+3
ConvertRowNo      .Equ          ConvertColNo+3
DoMWSubr          .Equ          ConvertRowNo+3
DumpCRSPT         .Equ          DoMWSubr+3        ; Need to load VC file
DumpFerShur       .Equ          DumpCRSPT+3
HSCellFrBS        .Equ          DumpFerShur+3
LoadCRSPT         .Equ          HSCellFrBS+3
PointAtRow        .Equ          LoadCRSPT+3
Sane2Args         .Equ          PointAtRow+3
SetBotRt          .Equ          Sane2Args+3
SwapViews         .Equ          SetBotRt+3

AOldStr           .Equ          SwapViews+3
LFormTable        .Equ          AOldStr+2
VFormTable        .Equ          LFormTable+5
PostFCF           .Equ          VFormTable+7
FuncNames         .Equ          PostFCF+3
```

-- end of file.40

--- FILE.39 ---

Spreadsheet Main Overlay Entry Points

```
; Book of references to Main portion of Calculator Edit

AnchorBox       .Equ            LSSSegs+2+3
CalcBHor        .Equ            AnchorBox+3
CalcBVer        .Equ            CalcBHor+3
CenterCurRow    .Equ            CalcBVer+3
CExt2Int        .Equ            CenterCurRow+3
CInt2Ext        .Equ            CExt2Int+3
ClearRow        .Equ            CInt2Ext+3
ConvFPD2S       .Equ            ClearRow+3

DispAll         .Equ            ConvFPD2S+3
DispCur         .Equ            DispAll+3
DispPrev        .Equ            DispCur+3
DispRow         .Equ            DispPrev+3
DidWinScr       .Equ            DispRow+3
DoSvSt          .Equ            DidWinScr+3
DoRetn2St       .Equ            DoSvSt+3
DOSync          .Equ            DoRetn2St+3
DrawIBox        .Equ            DoSync+3
DlEntry         .Equ            DrawIBox+3
ExpandCursor    .Equ            DlEntry+3           ; displays on screen
FlEntry         .Equ            ExpandCursor+3
GetSpecs        .Equ            FlEntry+3
MCur2Work       .Equ            GetSpecs+3
MCCBBRC         .Equ            MCur2Work+3
MCCBTLC         .Equ            MCCBBRC+3
MCCBTLR         .Equ            MCCBTLC+3
MCCBBRR         .Equ            MCCBTLR+3
MoveCursor      .Equ            MCCBBRR+3
SetCorners      .Equ            MoveCursor+3
UndispCur       .Equ            SetCorners+3

ScrInvStatus    .Equ            UndispCur+3

-- end of file.39
```

--- FILE.10 ---

Word Processor Common Work Area

```
; Common routines for Word Processor assembly

; Equates for Quick Writer
FirstDLine      .Equ            2
IntReturn       .Equ            0D0             ; Internal rep of Return.
MainSegNo       .Equ            16.             ; Main is seg.16.
Task            .Equ            "W"             ; ie., Quick File.

                .If             Opsys=2
VisibleCR       .Equ            7F              ; End of line in zoom in.
                .Endc

; Zero page addresses for Quick Writer
CurSvSw         .Equ    ZP+0A8                  ; Boolean, cursor has to be found.
LastVer         .Equ    ZP+0AA                  ; Last screen line available for
                                                ; text.  Usually 21.
; The next two indicate what screen lines are to be displayed by routines.
DispFrom        .Equ    ZP+0AB                  ;
DispTo          .Equ    ZP+0AC
DispOver        .Equ    ZP+0AD                  ; Boolean.  Must display over previous
                                                ; data, whatever it was.
DispLn22        .Equ    ZP+0AE                  ; Boolean
DispLn23        .Equ    ZP+0AF

; The next are pointers within the data structures
CurAdDSP        .Equ    ZP+0B0                  ; Address of DSP corresponding to
                                                ; current cursor location.  Not
                                                ; always current
PrevAdDSP       .Equ    ZP+0B2

;               Zero page information about current data line
;               First for text lines
DLStickies      .Equ    ZP+0B4
DLLength        .Equ    ZP+0B5      ; 01-50 is text
DLCRBit         .Equ    ZP+0B6

;               In case data line is command or carriage return line
DLCount         .Equ    ZP+0B4
DLCommand       .Equ    ZP+0B5      ; D0+ is command

                .If             Language="A"
ComBold         .Equ            2               ; Control B
ComCopy         .Equ            "C"+0AKey
ComDelete       .Equ            "D"+0AKey
ComFind         .Equ            "F"+0AKey
ComCalc         .Equ            "K"+0AKey
ComMove         .Equ            "M"+0AKey
ComChName       .Equ            "N"+0AKey
ComPrOptions    .Equ            "O"+0AKey
ComPrint        .Equ            "P"+0AKey
ComReplace      .Equ            "R"+0AKey
ComSticky       .Equ            " "+0AKey
ComTabs         .Equ            "T"+0AKey
ComUnderLine    .Equ            12.             ; Control L
ComZoom         .Equ            "Z"+0AKey
ComPageBack     .Equ            11.+0AKey
ComPageForw     .Equ            10.+0AKey
                .Endc
```

```
                .If             Language="F"
ComBold         .Equ            7                       ; Control G
ComCopy         .Equ            "C"+0AKey
ComDelete       .Equ            "E"+0AKey
ComFind         .Equ            "L"+0AKey
ComCalc         .Equ            "K"+0AKey
ComMove         .Equ            "I"+0AKey
ComChName       .Equ            "N"+0AKey
ComPrOptions    .Equ            "O"+0AKey
ComPrint        .Equ            "P"+0AKey
ComReplace      .Equ            "R"+0AKey
ComSticky       .Equ            " "+0AKey
ComTabs         .Equ            "M"+0AKey
ComUnderLine    .Equ            19.                     ; Control S
ComZoom         .Equ            "Z"+0AKey
ComPageBack     .Equ            11.+0AKey
ComPageForw     .Equ            10.+0AKey
                .Endc

                .If             Language="G"
ComBold         .Equ            6
ComCopy         .Equ            "K"+0AKey
ComDelete       .Equ            "L"+0AKey
ComFind         .Equ            "F"+0AKey
ComCalc         .Equ            "U"+0AKey
ComMove         .Equ            "B"+0AKey
ComChName       .Equ            "N"+0AKey
ComPrOptions    .Equ            "P"+0AKey
ComPrint        .Equ            "D"+0AKey
ComReplace      .Equ            "E"+0AKey
ComTabs         .Equ            "T"+0AKey
ComZoom         .Equ            "A"+0AKey
ComPageBack     .Equ            11.+0AKey
ComPageForw     .Equ            10.+0AKey
ComSticky       .Equ            " "+0AKey
ComUnderline    .Equ            1f
                .Endc

                .If             Language="I"
ComBold         .Equ            14.                     ; Control N
ComCopy         .Equ            "O"+0AKey
ComDelete       .Equ            "E"+0AKey
ComFind         .Equ            "C"+0AKey
ComCalc         .Equ            "K"+0AKey
ComMove         .Equ            "M"+0AKey
ComChName       .Equ            "N"+0AKey
ComPrOptions    .Equ            "P"+0AKey
ComPrint        .Equ            "S"+0AKey
ComReplace      .Equ            "R"+0AKey
ComTabs         .Equ            "T"+0AKey
ComZoom         .Equ            "Z"+0AKey
ComPageBack     .Equ            11.+0AKey
ComPageForw     .Equ            10.+0AKey
ComSticky       .Equ            " "+0AKey
ComUnderline    .Equ            19.                     ; Control S
                .Endc
```

```
; Equates that define the printer options.  Makes them easier to change.
POPW            .Equ            0D8
POLM            .Equ            0D9
PORM            .Equ            0DA
POCI            .Equ            0DB
POP1            .Equ            0DC
POP2            .Equ            0DD
POIN            .Equ            0DE
POJU            .Equ            0DF
POUJ            .Equ            0E0
POCN            .Equ            0E1
POPL            .Equ            0E2
POTM            .Equ            0E3
POBM            .Equ            0E4
POLI            .Equ            0E5
POSS            .Equ            0E6             ; Single spacing
PODS            .Equ            0E7
POTS            .Equ            0E8
PONP            .Equ            0E9
POGB            .Equ            0EA
POGE            .Equ            0EB             ; Group
POHE            .Equ            0EC
POFO            .Equ            0ED             ; Footer
POSK            .Equ            0EE
POPN            .Equ            0EF
POPE            .Equ            0F0
POPH            .Equ            0F1
POSM            .Equ            0F2

;               These are not available on options menu.  They are inserted
;               when printing.
POPNPlus        .Equ            0F3
POPC1           .Equ            0F4     ; Page break, count is page number
POPC1Plus       .Equ            0F5     ; Page break, count+256 is page number

POPC2           .Equ            0F6     ; Page break, split a paragraph
POPC2Plus       .Equ            0F7     ; Page break, split a paragraph

;               Imbedded commands
POBB            .Equ            001
POBE            .Equ            002
POPlusB         .Equ            003
POPlusE         .Equ            004
POMinusB        .Equ            005
POMinusE        .Equ            006
POUB            .Equ            007
POUE            .Equ            008
POPP            .EQU            009
POEK            .Equ            00A
POStick         .Equ            00B
POMM            .Equ            00C     ; Out of order


; The following is the layout of the Quick Writer work area.  The beginning
; address is somewhat variable until this settles down.
                .List

; Information that is built while a file is open, and not important
; to save when closed

;               Leave a two byte buffer for mistakes
BldRecWA        .Equ            WPWA
BldRecText      .Equ            BldRecWA+1      ; Text portion

; One switch for each of first 22 lines on screen 0..21.  The value is the
; number of characters displayed the last time the line was done.  Values are
; from 1 to decimal 80.  $Hex 80 is a special case of a  carriage return display.
PrevDisp        .Equ            BldRecWA+100.

; A pointer to the DSP in DSPTab for each line on the screen.  Zeroes are valid,
; and indicate beyond the end of the document.
ScrAdDSP        .Equ            PrevDisp+24.
```

```
; Information about previous line.  Only used occasionally.
;                First for text lines.
PLStickies      .Equ            ScrAdDSP+48.
PLLength        .Equ            PLStickies+1
PLCRBit         .Equ            PLLength+1
PLRecWA         .Equ            PLCRBit+1       ; Length 82

;               In case data line is command or carriage return line
PLCount         .Equ            PLStickies
PLCommand       .Equ            PLCount+1

; How to find the cursor if its place is marked as so many bytes past the first
; byte of a certain DSP.
CurSvDSP        .Equ            PLRecWA+82.
CurSvCount      .Equ            CurSvDSP+2      ; 3 bytes bytes to skip past.
CurSvLM         .Equ            CurSvCount+3    ; In case forget, comm lines
only
CurSvPE         .Equ            CurSvLM+1       ; Bytes past last valid char.
CurSvAtLM       .Equ            CurSvPE+1       ; Boolean, was on LM at time
CurSvVer        .Equ            CurSvAtLM+1     ; Ver when key was pressed

; Request being made of segment.  Like #ComPrint
CallerComm      .Equ            CurSvVer+1

; Tell PutCurData to bitch about lost text.  Don't in Copy.
PCBitch         .Equ            CallerComm+1

; Beginning of the word processor master record.
WPHeader        .Equ            PCBitch+1
MSNextDSP       .Equ            WPHeader
IWFiller1       .Equ            MSNextDSP+2     ; ADDSP top line when left
TabStLen        .Equ            IWFiller1+2
TabStops        .Equ            TabStLen+1      ; "-" is no-stop, "!" is stop.
ZoomInSw        .Equ            TabStops+80.

; These parameters are used to return to the same screen contents whenever
; this file is selected from among those on the desktop.  Although saved on
; disk, they are re-initialized when a file is read from disk.
ResumeDSP       .Equ            ZoomInSw+1      ; Top line of screen
IWLastVer       .Equ            ResumeDSP+2
IWLastHor       .Equ            IWLastVer+1

;               Now some miscellaneous stuff
ValidPagin      .Equ            IWLastHor+1     ; Boolean, pagination is OK
MinLM           .Equ            ValidPagin+1    ; Min Left Mar whole doc.
MMInDoc         .Equ            MinLM+1         ; Boolean, mail merge in file

; Work area to store record pointers.  Values higher than D000 are commands.
; Variable number of records, depending on whether Slinky here or not.
;               First text line is right at ARecords.
ARecords        .Equ            WPHeader+300.

-- End of file.10
```

--- FILE.37 ---

Word Processor Main Segment Entry Points

```
ClearFileHeader  .Equ    LWPMain+2+3
CurVsEnd         .Equ    ClearFileHeader+3
CurVsLast        .Equ    CurVsEnd+3
CurVsNext        .Equ    CurVsLast+3
MsgMaxLines      .Equ    CurVsNext+3
NoMMDMsg         .Equ    MsgMaxLines+3
PointAtOptions   .Equ    NoMMDMsg+3
PutCurData       .Equ    PointAtOptions+3
RelAn82          .Equ    PutCurData+3
SetCurData       .Equ    RelAn82+3
TextLost         .Equ    SetCurData+3
VerBotLine       .Equ    TextLost+3
PCPointers       .Equ    VerBotLine+3
ICPointers       .Equ    PCPointers+216.
```

-- end of file.37

--- FILE.2 ---

Part I of File formats for AppleWorks and /// E-Z Pieces

This document describes the file formats for AppleWorks and /// E-Z Pieces as of 11/20/86. This is not confidential information, and may be copied as necessary.

Definitions

LSB: The least significant byte of a word. It is also the "left byte," the "first byte off the disk," and the "lower address in memory." As you can see, we're going to try to explain our buzzwords.

MSB: The most significant byte of a word. Its address will be one higher than the address of the LSB, in memory, and on disk. If the total value of a word is less than 256D, this byte will be zero.

MRL: Data base multiple record layout.

SRL: Data base single record layout.

RAC: Review/Add/Change screen

1W: Shorthand for "one word." You will also see references to bytes, in the form "3B," for example.

1BI: One byte integer. Values from 0 to 255 decimal. A value of decimal 50 will be stored as 32H.

1WI: One word integer. Bit 80 of the LSB has a value of 128 decimal. Bit 80 of the MSB is the sign bit. Bit 01 of the MSB has a value of 256 decimal.

Boolean: A single byte that either zero ("false") or has the rightmost bit turned on ("true"). If the rightmost bit is on ("true") there may also be other bits on. These additional bits have no meaning.

String: A 1BI followed by the number of characters shown in the value of the 1BI.

ProDOS: All references to ProDOS apply equally to SOS.

DB: AppleWorks and/or /// E-Z Pieces Data Base

SS: AppleWorks and/or /// E-Z Pieces Spreadsheet

WP: AppleWorks and/or /// E-Z Pieces Word Processor

AW: AppleWorks and/or /// E-Z Pieces

Bit 80: All bits are referred to by their hexidecimal value. 80 is the leftmost bit in a byte, 01 is the rightmost.

"+" All addresses within an area will be relative to the first byte of the area. For example, the first byte of the area will be referred to as "Byte +0," the 10th byte would be "Byte +9."

Hex/dec: All hexadecimal numbers will be shown with a trailing H as in 1CH. Decimal numbers may sometimes be shown with a trailing D. Numbers without a D or an H are decimal.

Disk Directory Information

You may wish to refer to Apple's SOS Reference Manual, Volume 1, page 100.

Three file types are reserved for AW files:

File type 19H	Data Base
File type 1AH	Word Processor
File type 1BH	Spreadsheet

The volume/subdirectory auxiliary type word (+1FH) is used by AW to control upper/lower case display of file names. Bit 80 of the LSB refers to the first character of the file name, bit 02 of the MSB refers to the 15th character, and so on.

AW performs the following steps when it saves one of its files to disk:

1. Zero all 16 bits of the auxiliary type word.
2. Examine the file name for lower case letters. If one is found, change to "1" the corresponding bit in auxiliary type word and change the letter to upper case.
3. Examine the file name for spaces. If a space is found, change to "1' the corresponding bit auxiliary type word, and change the space to a period.

When files are read from disk, the file name and auxiliary type information from the directory "file entry" are used to determine which characters should be lower case, and which periods should be displayed as spaces.

If you use the auxiliary type bytes for a different purpose, AW will still display the file names, but it is likely that the wrong letters will be lower case.

Data Base files start with a variable length header, followed by 600 bytes for each report format (if any), the "standard values" record, and then variable length information for each "record."

Header Record

The header contains category names, record selection rules, counts, screen positioning information and all other information that is not specific to one record.

+000 - +001	1WI that is the number of bytes in the remainder of the header record. Use this count for your next ProDOS read from the disk.
+002 - +029	Ignore these bytes.
+030	Cursor direction when Return is pressed in SRL. 1H: Order in which you defined categories, or 2H: Left to right, top to bottom.
+031	What direction should the cursor go when you press Return in the MRL? D)own or R)ight.
+032 - +033	Ignore these bytes.
+034	Style of display that Review/Add/Change was using when the file was saved: R: SRL. /: MRL.
+035	1BI Number of categories per record. Values from 01H to 1EH
+036 - +037	1WI Number of records in file.
+038	1BI number of reports in this file, maximum of 8.
+039 - +041	Ignore these bytes.
+042 - +071	1BI for each of up to 30 columns, showing the number of spaces used for this column on the MRL. Be sure that you understand that categories may have categories on the MRL, the defined category that appears in each position. Byte +078 is the leftmost column of the MRL, and has a value from 01H to 1EH that defines which of the category names appears in this position. These numbers change as a result of changing the layout of the MRL.
+108 - +113	Ignore six bytes.
+114 - +143	1BI For up to 30 categories on the SRL, the horizontal screen position. These are changed as a result of changing the layout of the SRL. AW makes sure that these entries, and the vertical screen positions, are kept in order from left to right within top to bottom.
+144 - +149	Ignore these six bytes.
+150 - +179	1BI For up to 30 categories on the SRL, the vertical screen position.
+180 - +185	Ignore six bytes.
+186 - +215	1BI For up to 30 categories on the SRL, which of the category names appears in this position. These change as a result of changing the SRL. This number refers to the category names listed below.
+216 - +221	Ignore six bytes.

-- continued next page --

+222 1BI Number of categories on MRL. Will be less than, or equal to the number of categories in the file. SRL displays all categories, so there is no equivalent number for SRL.

+223 - +224 1WI For first line of RAC selection rules. Zero means no selection rules, any other value refers to the category name that is tested. MSB will always be zero.

+225 - +226 1WI Category name for second line of RAC selection rules. Zero means that there is only one line.

+227 - +228 1WI. Category name for third line of RAC selection rules. Zero means that there is no third line.

+229 - +230 1WI. For first line of RAC rules, which of the tests is to be applied. 1 means "equals," 2 means "greater than" and so on.

+231 - +232 Test for second line of rules, if any.

+233 - +234 Test for third line, if any.

+235 - +236 1WI Continuation code for first line: 1: And, 2: Or, 3: Through.

+237 - +238 1WI Continuation code for second line.

+239 - +240 1WI Continuation code for third line. Not possible, so always zero.

+241 - +272 String, max length 30 bytes. Comparison information for first line RAC selection rules.

+273 - +304 Comparison for 2nd line.

+305 - +336 Comparison for 3rd line.

+337 - +356 Ignore these six bytes.

+357 - +378 Name of the first category. A string, maximum length 20 bytes. If the file has only one category, the header record will end here.

+379 - +400 Name of the second category, if any. This area will not be on the header record if there is only one category.

+401... Additional 22 byte entries for all remaining categories. The size of the header record depends on the number of categories. Space is not maintained past the last category.

Report Records

Report Records follow the header record. One of the header record categories tells you how many report records to expect. The number will be from 0 to 8. Each report record is 600 bytes, and contains:

+000 - +019 String (max 19) Report name

+020 - +052 Column width (1BI) for up to 33 columns a tables style report format. Byte +020 is for the leftmost column on a tables style report. There can be up to 30 categories from the file, plus 3 more calculated columns.

For labels style report formats, the value is a 1BI that has the horizontal position of this category, relative to the left margin.

+053 - +055 Skip 3 bytes.

+056 - +088 For tables style: Number of spaces to be printed at the right of justified columns.

For labels style: Vertical position on the report for each of up to 30 categories. 1BI. A value of 1 means that category is on first line of labels style report.

+089 - +091 Skip 3 bytes.

+092 - +124 For up to 33 columns of tables style: Values from 1 to 30D refer to which category name appears in this column on the report. Values of 80H, 81H and 82H are the three calculated categories, from left to right.

For labels style: Same as tables style, minus the calculated categories.

+125 - +127 Skip these three columns

+128 - +160 For up to 33 columns of tables style: 99H means no foot totals, 0 thru 4 means the number of decimal places for a foot total.

For labels style: For up to 30 categories on report, boolean whether or not category names are to be printed.

+161 - +163 Skip these three bytes.

+164 - +196 For up to 33 columns of tables style: 99H means left justified, 0H through 4H means right justified with 0 to 4 decimal places.

For up to 30 categories of labels style: Boolean whether or not to float (0A-J) this category up against the category to its left.

+197 - +199 Skip three bytes.

+200 1BI. Number of categories on report. Includes calculated categories, if any.

+201 Tables style. If there is at least one calculated category, this is a 1BI containing values from 1H to 33D: which column of the report.

Labels style: 1BI. Values from 3H to 21D. Position of line on the screen that says "Each record will print nn lines."

+202 Tables style: Same as +201, but for the second calculated category, if any.

Labels style: Unused.

-- continued next page --

+203 Tables style: Same as +201, but for the third calculated category, if any.

Labels style: Unused

+204 Tables style only: If there is a group total column, a 1BI stating which of the category names is used as a basis. Values from 1D to 30D.

+205 Platen width value, in 10ths of an inch. For example, a value of 8.0 inches entered by the user will show as 80D or 50H.

+206 Left margin value. All inches values are in 10ths.

+207 Right margin value.

+208 Characters per inch. 1BI.

+209 Paper length value. 10ths of an inch.

+210 Top margin value.

+211 Bottom margin value.

+212 Lines per inch. 6H or 8H. 1BI.

+213 Not relevant. Probably always a "C".

+214 Type of report format. H: tables style, V: labels style.

+215 Spacing: S(ingle, D(ouble, or T(riple. Expect these three letters, even in European versions.

+216 Print report header. Boolean.

+217 Tables style: If user has specified group totals, Boolean: just print the group totals.

+218 Labels style: Boolean, omit line when all entries on line are blank.

+219 Labels style: Boolean, keep number of lines the Same within each record.

+220 - +301 80 byte string. Title line, if any.

+302 - +323 Tables style. 20 byte string. Name of first calculated category, if any.

+324 - +355 Tables style. 30 byte string. Calculation rules for first calculated category, if any.

+356 - +409 Tables style. Name and rules for second calculated category, if any.

+410 - +463 Tables style. Name and rules for third calculated category, if any.

+464 - +477 If user has specified "Send special codes to printer," this is a 13 byte string containing those codes.

+478 Boolean: Print a dash when an entry is blank.

+479 - +592 Record selection rules. Exact same format as described in the header record.

+593 - +599 Unused

Data Records

Data records follow the report records. The first data record contains the standard values. Each following data record corresponds to one data base "record."

These records contain all of the categories within one stream of data. The category entries are in the same order that the category names appear in the header record.

Bytes +0 and +1 are a Word that contains a count of the number of bytes in the remainder of the record.

Byte +2 of each record will always be a control byte. Other control bytes within each record define the contents of the record. Control bytes may be:

01-7FH	This is a count of the number of following bytes that are the contents of a category.
81-9EH	This (minus 80) is a count of the number of categories to be skipped. For example, 82H means skip two categories.
FFH	This indicates the end of the record.

The information in individual categories may have some special coding so that date and time entries can be arranged (sorted).

Date entries will have the following format:

+000	C0H (192D). Identifies a date entry.
+001 - 002	Ascii year code, like 84.
+003	Ascii month code. A means January, L means December.
+004 - +005	Ascii day of the month.

Time entries will have the following format:

+000	D4H (212D). Identifies a time entry.
+001	Ascii hour code. A means 00 (the hour after midnight). X means 23, the hour before midnight.
+002 - +003	Ascii minute code. Values from 00 to 59.

This is the end of the data base file formats.

Word Processor Files

Word Processor files start with a 300 byte header, followed by a number of variable length line records, one for each line on the screen.

Header Records

The header contains the following information:

+000 - +003	Not used.
+004	4FH (79D)
+005 - +084	Tab stops. Either = or !.
+085	Zoom switch. Boolean.
+086 - +089	Four bytes not used.
+090	Boolean, whether file is currently paginated, ie., whether the page break lines are showing.
+091	Minimum left margin that should be added to the margin that is appearing on the screen. This is normally one inch, shown in 10ths of an inch, 10D or 0AH.
+092	Boolean, whether file contains any mail-merge commands.
+093 - +249	Not used. More or less reserved for "mail merge" feature.
+250 - +299	Available. Will Never be used by AppleWorks. If you are creating AW WP files, you can use this area to keep information that is important to your program.

Line Records

Line records are of three different types. The first line record after the 300 byte header corresponds to line 1, the next is line 2, and so on. The first two bytes of each line record contain enough information to establish the type.

Carriage return line records have a D0H (208D) in byte +001. Byte +000 is a 1BI between 00 and 79D that is the horizontal screen position of this carriage return.

Command line records are formatting commands that appear on the screen in the form ~~~~~~~~Double Space, for example. These can be identified by a value greater than D0H (208D) in byte +001. They are:

Byte +001	Command	Byte +000
D8H	Platen width	10ths of an inch
D9H	Left margin	10ths of an inch
DAH	Right margin	10ths of an inch
DBH	Chars per inch	1BI.
DCH	Proportional-1	No meaning
DDH	Proportional-2	
DEH	Indent	1BI characters
DFH	Justify	
E0H	Unjustify	
E1H	Center	
E2H	Paper length	10ths of an inch
E3H	Top margin	10ths of an inch
E4H	Bottom margin	10ths of an inch
E5H	Lines per inch	1BI.
E6H	Single space	
E7H	Double space	
E8H	Triple space	
E9H	New page	
EAH	Group begin	
EBH	Group end	
ECH	Page header	
EDH	Page footer	

-- continued next page --

EEH	Skip lines	1BI count
EFH	Page number	1BI
F0H	Pause each page	
F1H	Pause here	
F2H	Set marker	1BI marker number
F3H	Page number	1BI (add 256)
F4H	Page break	1BI page number
F5H	Page break	1BI (add 256)
F6H	Page break	1BI (break in middle of paragraph)
F7H	Page break	1BI (add 256, in middle of paragraph)
FFH	End of file	

Text Records

Text records are the lines where text has been typed. The format is:

+000 - +001 Word. Number of bytes following this word. Since the maximum is about 80D, byte +001 is always zero. Use byte +001 to identify text lines.

+002 Screen column for the first text character. Usually will be zero, but may vary as a result of left margin, centering, and indent commands.

+003 80 bit: If on, there is a carriage return on the end of this line. If off, no carriage return.

Remaining 7 bits: Number of bytes of text following this byte.

+004 - nnn Actual text bytes. Consists of Ascii characters and special codes. The special codes are values from 01H to 1FH, and indicate special formatting features:

Code	Meaning
01H	Begin boldface
02H	Boldface end
03H	Superscript begin
04H	Superscript end
05H	Subscript begin
06H	Subscript end
07H	Underline begin
08H	Underline end
09H	Print page number
0AH	Enter keyboard
0BH	Sticky space
0CH	Mail merge

This is the end of word processor file format.

-- End of file.2

Spreadsheet Files

Spreadsheet files start with a 300 byte header record that contains basic information about the file, including column widths, printer options, window definitions, and standard values.

Header Record

The spreadsheet header record contains the following entries:

+000 - +003	Skip 4 bytes.
+004 - +130	1BI containing the column width for each column.
+131	Order of recalculation. ASCII R or C.
+132	Frequency of recalculation. ASCII A or M.
+133 - +134	Last row referenced. Word.
+135	Last column referenced. 1BI.
+136	Number of windows: ASCII 1: just one window, S: side by side windows, T: top and bottom windows.
+137	Boolean. If there are two windows, are they synchonized?
+138 - +161	The next 20 (approx) variables are for the current window. If there is only one window, it is the current window. If there are two windows, the current window is the window that had the cursor in it.
+138	Window standard format for label cells. 2H: left justified, 3H: right justified, 4H: centered.
+139	Window standard format for value cells. 2H: fixed, 3H: dollars, 4H: commas, 5H: percent, 6H: appropriate
+140	More of window standard format for value cells. Number of decimal places to display. Values from 0 to 7H.
+141	Top screen line used by this window. This is the line that the ====A====B==== appears on. Normally 1H unless there are top and bottom windows.
+142	Leftmost screen column used by this window. This is the column that the hundreds digit of the row number appears in. Normally 0H unless there are side by side windows.
+143 - +144	Top, or first, row appearing in titles area. This will probably be zero if there are no "top" titles.
+145	Leftmost, or first, column appearing in left side titles area. This will probably be zero if there are no left side titles.
+146 - +147	Last row appearing in top titles area. This will probably be zero if there are no top titles.
+148	Last column appearing in left side titles area. This will probably be zero if there are no left side titles.
+149 - +150	Top, or first, row appearing in the body of the window. The body is defined as those rows that are on the screen, but not in the titles area.

-- continued next page --

+151	Leftmost, or first, column appearing in the body of the window.
+152	The screen line that the top body row goes on. Normally 2H unless there are top titles, or top and bottom windows.
+153	Leftmost screen column used for the leftmost body column. Normally 4H unless there are side titles, or side by side windows.
+154 - +155	Bottommost, or last, row appearing in this window.
+156	Rightmost, or last, column appearing in this window.
+157	The screen line that the last body row goes on. Normally 13H (19D) unless there are top and bottom windows.
+158	The rightmost screen column used by this window. Normally 4EH (78D) unless there are side by side windows.
+159	1BI. Number of horizontal screen locations used to display the body columns. Normally 48H (72D), because 8 columns of 9 characters each are the standard display. This is effected by side by side windows, side titles, and variable column widths.
+160	Boolean. Rightmost column is not fully displayed. This can only happen when the body portion of the window is narrower than the width of a particular column.
+161	Titles switch for this window. Bit 80: top titles, Bit 40: side titles. These bits represent top titles, side titles, both, and no titles.
+162 - +185	Window information for the secondary window. This is meaningful only if there are two windows. This is the information for the window that the cursor is not currently in. See the descriptions for the current window (+138 - +161).
+186 - +212	Not currently used.
+213	Boolean, cell protection is on or off.
+214	Not currently used.
+215	Platen width value, in 10ths of an inch. For example, a value of 8.0 inches entered by the user will show as 80D or 50H.
+216	Left margin value. All inches values are in 10ths.
+217	Right margin value.
+218	Characters per inch. 1BI.
+219	Paper length value. 10ths of an inch.
+220	Top margin value.
+221	Bottom margin value.
+222	Lines per inch. 6H or 8H. 1BI.
+223	Spacing: S(ingle, D(ouble, or T(riple. Expect these three letters, even in European versions.
+224 - +237	If user has specified "Send special codes to printer," this is a 13 byte string containing those codes.
+238	Boolean: Print a dash when an entry is blank.
+239	Print report header. Boolean.
+240	Boolean. Zoomed in to show formulas.
+241 - +249	Reserved for future use.
+250 - +299	Available. Will Never be used by AppleWorks. If you are creating AW 88 files, you can use this area to keep information that is important to your program.

Row Records

Row records contain a variable amount of information about each row that is non-blank. Each row record contains enough information to completely build one row of the spreadsheet:

+000 - +001 Number of additional bytes to read from disk. FFFFH means end of file.

+002 - +003 Word. Row number.

+004 Beginning of actual information for the row. This byte of each record will always be a control byte. Other control bytes within each record define the contents of the record. Control bytes may be:

01-7FH This is a count of the number of following bytes that are the contents of a cell entry.

81-FEH This (minus 80) is a count of the number of columns to be skipped. For example, 82H means skip two columns.

FFH This indicates the end of the row.

Cell Entries

Cell entries contain all the information that is necessary to build one cell. There are several types:

Value constants are cells that have a value that cannot change. This means that someone typed a constant into the cell, 3.14159, for example.

+000 Bit 80 is always on.

Bit 40 on means that if the value is zero, display a blank instead of a zero. This is for pre-formatted cells that still have no value.

Bit 20 is always on.

Bit 10 on means that labels cannot be typed into this cell.

Bit 08 on means that values cannot be typed into this cell.

Bits 04, 02, and 01 specify the formatting for this cell:

1: Use spreadsheet standard
2: Fixed
3: Dollars
4: Commas
5: Percent
6: Appropriate

+001 Bit 80: always zero.

Bit 40: always zero.

Bit 20: always zero.

Bit 10: On indicates that this cell must be calculated the next time this spreadsheet is calculated, even if none of the referenced cells are changed. This bit makes sense on for cells that have a calculated formula.

Bits 04, 02, 01. Number of decimal places for fixed, dollars, commas, or percent formats.

+002 - +009 8 byte Apple SANE numerics double format floating point number.

Value formulas are cells that contain information that has to be evaluated. Formulas like AA17+@sum(r19...r21), and @Error are examples. Specific format:

+000	Bit 80 is always on. Bit 40 on means to not display the cell. This was originally intended for pre-formatted cells that still have no value. If a value is placed in this cell, be sure to turn off this bit. Bit 20 is always off. Bits 10, 08, 04, 02, and 01 are the same as value constants.
+001	Bit 80: always on. Bit 40: On indicates that the last evaluation of this formula resulted in an @NA. Bit 20: On indicates that the last evaluation of this formula resulted in an @Error. Bits 10, 04, 02, 01 same as value constants.
+002 - +009	8 byte SANE double floating point number that is the most recent evaluation of this cell.
+010 - nnn	Various control bytes that are "tokens" representing the formula that was typed in by the user. They are:

Byte	Means
D9H	@Round
DAH	@Or
DBH	@And
DCH	@Sum
DDH	@Avg
DEH	@Choose
DFH	@Count
E0H	@Error (followed by 3 bytes of zero)
E1H	Reserved for future @IRR
E2H	@If
E3H	@Int
E4H	@Lookup
E5H	@Max
E6H	@Min
E7H	@NA (followed by three bytes of zero)
E8H	@NPV
E9H	@Sqrt
EAH	@Abs
EBH	Not currently use
ECH	<> (not equal)
EDH	>= (greater than or equal to)
EEH	<= (less than or equal to)
EFH	= (equals)
F0H	> (greater than)
F1H	< (less than)
F2H	, (comma)
F3H	^ (exponentiation sign)
F4H	) (right parenthesis)
F5H	- (minus)
F6H	+ (plus)
F7H	/ (divide)
F8H	* (multiply)
F9H	((left parenthesis)
FAH	- (unary minus. Ex: -A3)
FBH	+ (unary plus. Ex: +A3)
FCH	... (dots)
FDH	Next 8 bytes are SANE double number
FEH	Next 3 bytes are row, column reference.

Two of the codes require special information. Code FDH indicates that the next 8 bytes are a SANE numerics package "double" precision floating point number. All constants within formulas are carried in this manner.

Code FEH indicates that the next three bytes point at a cell:

+000	FEH
+001	Column reference. Add this 1BI to the column number of the current cell to get the column number of the pointed at cell. This value is sometimes negative, but Add always works.
+002 - +003	Row reference. Word. Add this word to the row number of the current cell to get the row number of the pointed at cell. This value is sometimes negative, but Add always works.

Propagated label cells are labels that place one particular ASCII character in each position of a window. Handy for underlining.

+000	Bit 80 is always zero. Bit 40 is meaningless. Bit 20 is always on. Bit 10 and bit 08 are protection, just like value cells. Bits 04, 02, 01 are meaningless. Put a 1 here.
+001	This is the actual character that is to be put in each position in the cell.

Regular label cells contain alphanumeric information, such as headings, names and other descriptive information that is not the basis for calculations.

+000	Bits 80, 40, 20 are always zero. Bits 10 and 08 are same as value cells. Bits 04, 02, and 01 determine cell formatting: 01: Use spreadsheet standard formatting 02: Left justify 03: Right justify 04: Center
+001 - +nnn	ASCII characters that actually display. The actual length was defined earlier in the word that contained the "actual number of bytes to read from disk."

This is the end of the spreadsheet file format.

This is the end of AppleWorks file formats. Your comments, in writing, would be appreciated. You can write to either:

Apple Computer, Inc.
AppleWorks Product Manager
20525 Mariani Avenue
Cupertino CA 95014

-- End of file.15

Printer Definitions Work Area

```
; This becomes segment Seg.PR.  It is definitions of printers.  All of the
; addresses are relative to zero, so all word addresses will have to be
; relocated when loaded.

                .Include        /p/iall/common
                .Include        /p/iall/Macros


                .List
                .Proc           PRDefs
                .Org            0


; So other segments can get past the environment record on disk.  Start out
; with enough bytes to store environment record.
                .Byte           0E0                     ; HardPathName
                String          07,"Drive 2"            ; MainPathName
                .Block          32.-8                   ; Remainder of MainPathName
                .Block          4                       ; Various

                .If             Language="A"
                String          8,"10/01/85"
                .Endc

                .If             Language="F"
                String          8,"20/05/86"
                .Endc

                .If             Language="G"
                String          8,"30.09.85"
                .Endc

                .If             Language="I"
                String          8,"30/09/85"
                .Endc

                .Block          10.                     ; Filler
                .Block          EREELength
                .Block          EREELength
                .Block          EREELength

; Now space for 768 bytes that come from D100 thru D3FF when we load.
                .Block          300

                .Include        /p/iall/segpr1

; Table of number of dots for Apple DMP proportional characters.

                .If             Language="A"
FATWidths       .Byte           07.+80,07.+80,10.,14.                   ; 20-23
                .Byte           12.,16.+80,13.,07.                      ; 24-27
                .Byte           07.,07.,12.,12.                         ; 28-2B
                .Byte           07.+80,12.,07.+80,12.                   ; 2C-2F

                .Byte           12.,12.,12.,12.                         ; 30-33
                .Byte           12.,12.,12.,12.                         ; 34-37
                .Byte           12.,12.,07.+80,07.+80                   ; 38-3B
                .Byte           12.,12.,12.,12.+80                      ; 3C-3F

                .Byte           14.,16.,15.,14.,15.,15.,15.,14.         ; 40-47
                .Byte           15.,09.,13.,12.,13.,17.,16.,15.         ; 48-4F

                .Byte           13.,16.,15.,12.,14.,15.,16.,17.         ; 50-57
                .Byte           11.,14.,11.,12.,12.,12.,12.,17.         ; 58-5F

                .Byte           07.,12.,12.,10.,12.,12.,10.,12.         ; 60-67
                .Byte           12.,08.,07.,10.,08.,16.,12.,12.         ; 68-6F

                .Byte           12.,12.,10.,12.,10.,12.,12.,16.         ; 70-77
                .Byte           12.,12.,10.,10.,07.,10.,13.,00.         ; 78-7F
                .Endc
```

```
                .If             Language="F"
PATWidths       .Byte           07.+80,07.+80,10.,13.                   ; 20-23
                .Byte           12.,16.+80,13.,07.                      ; 24-27
                .Byte           07.,07.,12.,12.                         ; 28-2B
                .Byte           07.+80,12.,07.+80,12.                   ; 2C-2F

                .Byte           12.,12.,12.,12.                         ; 30-33
                .Byte           12.,12.,12.,12.                         ; 34-37
                .Byte           12.,12.,07.+80,07.+80                   ; 38-3B
                .Byte           12.,12.,12.,12.+80                      ; 3C-3F

                .Byte           12.,16.,15.,14.,15.,15.,15.,14.         ; 40-47
                .Byte           15.,09.,13.,12.,13.,17.,16.,15.         ; 48-4F

                .Byte           13.,16.,15.,12.,14.,15.,16.,17.         ; 50-57
                .Byte           11.,14.,11.,13.,10.,12.,12.,17.         ; 58-5F

                .Byte           07.,12.,12.,10.,12.,12.,10.,12.         ; 60-67
                .Byte           12.,08.,07.,10.,08.,16.,12.,12.         ; 68-6F

                .Byte           12.,12.,10.,12.,10.,12.,12.,16.         ; 70-77
                .Byte           12.,12.,16.,12.,12.,12.,13.,00.         ; 78-7F
                .Endc

                .If             Language="G"
PATWidths       .Byte           07.+80,07.+80,10.,14.                   ; 20-23
                .Byte           12.,16.+80,13.,07.                      ; 24-27
                .Byte           07.,07.,12.,12.                         ; 28-2B
                .Byte           07.+80,12.,07.+80,12.                   ; 2C-2F

                .Byte           12.,12.,12.,12.                         ; 30-33
                .Byte           12.,12.,12.,12.                         ; 34-37
                .Byte           12.,12.,07.+80,07.+80                   ; 38-3B
                .Byte           12.,12.,12.,12.+80                      ; 3C-3F

                .Byte           12.,16.,15.,14.,15.,15.,15.,14.         ; 40-47
                .Byte           15.,09.,13.,12.,13.,17.,16.,15.         ; 48-4F

                .Byte           13.,16.,15.,12.,14.,15.,16.,17.         ; 50-57
                .Byte           11.,14.,11.,16.,15.,15.,12.,17.         ; 58-5F

                .Byte           07.,12.,12.,10.,12.,12.,10.,12.         ; 60-67
                .Byte           12.,08.,07.,10.,08.,16.,12.,12.         ; 68-6F

                .Byte           12.,12.,10.,12.,10.,12.,12.,16.         ; 70-77
                .Byte           12.,12.,10.,12.,12.,12.,14.,00.         ; 78-7F
                .Endc

                .If             Language="I"
PATWidths       .Byte           07.+80,07.+80,10.,13.                   ; 20-23
                .Byte           12.,16.+80,13.,07.                      ; 24-27
                .Byte           07.,07.,12.,12.                         ; 28-2B
                .Byte           07.+80,12.,07.+80,12.                   ; 2C-2F

                .Byte           12.,12.,12.,12.                         ; 30-33
                .Byte           12.,12.,12.,12.                         ; 34-37
                .Byte           12.,12.,07.+80,07.+80                   ; 38-3B
                .Byte           12.,12.,12.,12.+80                      ; 3C-3F

                .Byte           12.,16.,15.,14.,15.,15.,15.,14.         ; 40-47
                .Byte           15.,09.,13.,12.,13.,17.,16.,15.         ; 48-4F

                .Byte           13.,16.,15.,12.,14.,15.,16.,17.         ; 50-57
                .Byte           11.,14.,11.,13.,10.,12.,12.,17.         ; 58-5F

                .Byte           12.,12.,12.,10.,12.,12.,10.,12.         ; 60-67
                .Byte           12.,08.,07.,10.,08.,16.,12.,12.         ; 68-6F

                .Byte           12.,12.,10.,12.,10.,12.,12.,16.         ; 70-77
                .Byte           12.,12.,10.,12.,12.,12.,08.,00.         ; 78-7F
                .Endc
```

```
; Table of number of 120ths for Apple Letter Quality proportional characters.

; 80 means it is punctuation.
PCTWidths       .Byte           06.+80,10.+80,10.,10.           ; 20-23
                .Byte           10.,10.+80,12.,04.              ; 24-27
                .Byte           10.,10.,10.,10.                 ; 28-2B
                .Byte           10.+80,10.,10.+80,10.           ; 2C-2F

                .Byte           10.,10.,10.,10.                 ; 30-33
                .Byte           10.,10.,10.,10.                 ; 34-37
                .Byte           10.,10.,10.+80,10.+80           ; 38-3B
                .Byte           12.,10.,12.,10.+80              ; 3C-3F

                .Byte           10.,14.,14.,14.,14.,12.,12.,14. ; 40-47
                .Byte           14.,08.,10.,14.,12.,14.,14.,14. ; 48-4F

                .Byte           12.,14.,14.,12.,14.,14.,14.,14. ; 50-57
                .Byte           14.,14.,12.,10.,10.,10.,16.,12. ; 58-5F

                .Byte           10.,10.,12.,10.,12.,10.,08.,12. ; 60-67
                .Byte           12.,06.,06.,12.,06.,14.,12.,10. ; 68-6F

                .Byte           12.,12.,10.,10.,08.,12.,12.,14. ; 70-77
                .Byte           12.,12.,10.,10.,12.,08.,10.,10. ; 78-7F

; Table of number of dots for Epson FX proportional characters.
PFTWidths       .Byte           12.+80,05.+80,08.,12.           ; 20-23
                .Byte           12.,12.+80,12.,05.              ; 24-27
                .Byte           06.,06.,12.,12.                 ; 28-2B
                .Byte           07.+80,12.,06.+80,10.           ; 2C-2F

                .Byte           12.,08.,12.,12.                 ; 30-33
                .Byte           12.,12.,12.,12.                 ; 34-37
                .Byte           12.,12.,06.+80,06.+80           ; 38-3B
                .Byte           10.,12.,10.,12.+80              ; 3C-3F

                .Byte           12.,12.,12.,12.,12.,12.,12.,12. ; 40-47
                .Byte           12.,08.,11.,12.,12.,12.,12.,12. ; 48-4F

                .Byte           12.,12.,12.,12.,12.,12.,12.,12. ; 50-57
                .Byte           10.,12.,10.,08.,10.,08.,12.,12. ; 58-5F

                .Byte           05.,12.,11.,11.,11.,12.,10.,11. ; 60-67
                .Byte           11.,08.,09.,10.,08.,12.,11.,12. ; 68-6F

                .Byte           11.,11.,11.,12.,11.,12.,12.,12. ; 70-77
                .Byte           10.,12.,10.,09.,05.,07.,12.,00. ; 78-7F



VeryEnd         .Equ            *

                .End
```

-- End of part 1

-- end of file.11

Printer Definitions Work Area

Part II of seg.pr source code.

```
; Book SegPr1.  Beginning of printer tables.

; Point at tables for each of the printers.  Printer 1
; is whatever printer is number 1 on the pick your type menu
                .Word           0                       ; No type 0
                .Word           PrATable                ; Apple DMP
                .Word           PrATable                ; Apple ImageWriter
                .Word           PrCTable                ; Apple DWP
                .Word           PRDTable                ; Apple Silentype

                .Word           PRGTable                ; Epson MX
                .Word           PRHTable                ; Epson MX Graftrax
                .Word           PRITable                ; Epson RX
                .Word           PRFTable                ; Epson FX

                .Word           PrETable                ; Qume sprint 5
                .Word           PrWTable                ; Qume Sprint 11

                .Word           PrSTable                ; Scribe

                .Word           CustomTable             ; Customized.

; Dots per inch for Apple DMP.  (Printer #1)
PrATable        .Byte           0,160.,144.,0,72.,80.,96.,120.  ; 0-7.
                .Byte           136.,72.,80.,0,96.,0,0,120.     ; 8-15.
                .Byte           0,136.,0,0,0,0,0,0              ; 16.-23.
                .Byte           0

; Dots per character for valid characters 0 to 24.
                .Byte           0,0,0,0,16.,16.,16.,16.         ; 0-7
                .Byte           16.,8,8,0,8,0,0,8               ; 8-15.
                .Byte           0,8,0,0,0,0,0,0                 ; 16-23.
                .Byte           0                               ; 24.

; Addresses of command strings for each possible chars per inch
                .Word           0,0,0,0                         ; 0-3
                .Word           PAT4CPI-PrATable
                .Word           PAT5CPI-PrATable
                .Word           PAT6CPI-PrATable
                .Word           PAT7CPI-PrATable
                .Word           PAT8CPI-PrATable
                .Word           PAT9CPI-PrATable
                .Word           PAT10CPI-PrATable
                .Word           0                               ; 11
                .Word           PAT12CPI-PrATable
                .Word           0,0
                .Word           PAT15CPI-PrATable
                .Word           0
                .Word           PAT17CPI-PrATable               ; 17
                .Block          14.

; For DMP, strings for outputting various commands
                .Word           0                       ; Zero is Terry port commands
                .Word           $101-PrATable           ; BB
                .Word           $102-PrATable           ; BE
                .Word           $103-PrATable           ; PB
                .Word           $104-PrATable           ; PE
                .Word           $105-PrATable           ; MB 5
                .Word           $106-PrATable           ; ME
                .Word           $107-PrATable           ; UB
                .Word           $108-PrATable           ; UE
                .Word           $109-PrATable           ; 6LPI
                .Word           $110-PrATable           ; 8LPI          ; 10
                .Word           $112-PrATable           ; Reset
                .Word           $112-PrATable           ; Init
                .Word           0                       ; All CPI commands
                .Word           $118-PrATable           ; P1
                .Word           $119-PrATable           ; P2            ; 15.

                .Word           PATWidths               ; DMP widths
```

```
;               Method for underlining          0: none, see eg34
                .Byte           1

;               Can do fractional justification
                .Byte           True

$101            .Byte           2,ComEscape,"!"     ; turn on bold
$102            .Byte           2,ComEscape,22
$103            .Byte           7,ComEscape,"T","0","8"
                .Byte           ComEscape,"r"
                .Byte           0A
$104            .Byte           5,ComEscape,"f",0A,ComEscape,"A"
$105            .Byte           5,ComEscape,"T","0","8",0A
;  $106            .Byte           7,ComEscape,"r",0A              ; fractional LF
;    .Byte          ComEscape,"A",ComEscape,"f"
$106            .Byte           16.,ComEscape,"T","2","0"
                .Byte           ComEscape,"r",0A
                .Byte           ComEscape,"T","1","2"
                .Byte           ComEscape,"f",0A
                .Byte           ComEscape,"A"
$107            .Byte           2,ComEscape,"X"     ; underline on
$108            .Byte           2,ComEscape,"Y"     ; Underline end
$109            .Byte           4,ComEscape,"A",ComEscape,"f"       ; 6 lpi forward
$110            .Byte           4,ComEscape,"B",ComEscape,"f"       ; 8 lpi forward
$112            .Byte           8,ComEscape,"<",ComEscape,"N",ComEscape,"A"
                .Byte           ComEscape,"f"
$118            .Byte           3,0F,ComEscape,"P"
$119            .Byte           3,0F,ComEscape,"p"

; Actual strings to cause various chars per inch
PAT4CPI         .Byte           3,0E,ComEscape,"n"
PAT5CPI         .Byte           3,0E,ComEscape,"N"
PAT6CPI         .Byte           3,0E,ComEscape,"E"
PAT7CPI         .Byte           3,0E,ComEscape,"e"
PAT8CPI         .Byte           3,0E,ComEscape,"Q"
PAT9CPI         .Byte           3,0F,ComEscape,"n"
PAT10CPI        .Byte           3,0F,ComEscape,"N"
PAT12CPI        .Byte           3,0F,ComEscape,"E"
PAT15CPI        .Byte           3,0F,ComEscape,"q"
PAT17CPI        .Byte           3,0F,ComEscape,"Q"


; Dots per inch for Apple Scribe  (Printer #11)
PrSTable        :Byte           0,0,0,0,0,80.,0,0           ; 0-7
                .Byte           136.,0,80.,0,0,0,0,0        ; 8-15.
                .Byte           0,136.,0,0,0,0,0,0          ; 16.-23.
                .Byte           0

; Dots per character for valid charactrers 0 to 24.
                .Byte           0,0,0,0,0,16.,0,0           ; 0-7
                .Byte           16.,0,8,0,0,0,0,0           ; 8-15.
                .Byte           0,8,0,0,0,0,0,0             ; 16-23.
                .Byte           0                           ; 24.

; Addresses of command strings for each possible chars per inch
                .Word           0,0,0,0,0                   ; 0-4
                .Word           PST5CPI-PrSTable
                .Word           0,0                         ; 6-7
                .Word           PST8CPI-PrSTable
                .Word           0                           ; 9
                .Word           PST10CPI-PrSTable
                .Word           0,0,0,0,0,0                 ; 11-16
                .Word           PST17CPI-PrSTable           ; 17
                .Block          14.
```

-- continued next page --

```
; For DMP, strings for outputting various commands
                .Word           0                       ; Zero is Terry port commands
                .Word           0                       ; BE
                .Word           0                       ; BE
                .Word           $103-PrSTable           ; PE
                .Word           $104-PrSTable           ; PE
                .Word           $105-PrSTable           ; MB 5
                .Word           $104-PrSTable           ; ME
                .Word           $107-PrSTable           ; UB
                .Word           $108-PrSTable           ; UE
                .Word           $109-PrSTable           ; 6LPI
                .Word           $110-PrSTable           ; 8LPI          ; 10

                .If             Language="A"
                .Word           0
                .Word           0
                .Endc

                .If             Language<>"A"
                .Word           $112    ; Just get Europeans set to longer paper
                .Word           $112
                .Endc

                .Word           0                       ; All CPI commands
                .Word           0
                .Word           0

                .Word           0                       ; No proportional for Scribe

;               Method for underlining                  0: none, see eg34
                .Byte           1

;               Can do fractional justification
                .Byte           False

$103            .Byte           2,ComEscape,"x"
$104            .Byte           2,ComEscape,"z"
$105            .Byte           2,ComEscape,"y"

$107            .Byte           2,ComEscape,"X"         ; underline on
$108            .Byte           2,ComEscape,"Y"         ; Underline end
$109            .Byte           2,ComEscape,"A"         ; 6 lpi forward
$110            .Byte           4,ComEscape,"T","1","8"                 ; 8 lpi
forward

; Europeans have longer paper.

                .If             Language="F"
$112            .Byte           6,ComEscape
                .Ascii          "H1685"
                .Endc

                .If             Language="G"
$112            .Byte           6,ComEscape
                .Ascii          "H1728"
                .Endc

                .If             Language="I"
$112            .Byte           6,ComEscape
                .Ascii          "H1728"
                .Endc

; Actual strings to cause various chars per inch
PST5CPI         .Byte           3,0E,ComEscape,"N"
PST8CPI         .Byte           3,0E,ComEscape,"Q"
PST10CPI        .Byte           3,0F,ComEscape,"N"
PST17CPI        .Byte           3,0F,ComEscape,"Q"



; Hopefully, Sprint 11 is same as DWP
PrWTable        .Equ            *
```

-- continued next page --

```
; Dots per inch for Apple DWP.   (Printer #3)
PrCTable        .Byte           0,120.,120.,0,120.,120.,120.,120.,; 0-7
                .Byte           120.,120.,120.,120.,120.,120.,0,120.
                .Byte           0,120.,0,0,120.,0,0,0
                .Byte           120.

; 120ths per character for valid charactrers 0 to 24.
                .Byte           0,0,0,0,30.,24.,20.,17.                 ; 0-7
                .Byte           15.,13.,12.,11.,10.,9,0,8               ; 8-15. cpi
                .Byte           0,7,0,0,6,0,0,0                         ; 16-23. cpi
                .Byte           5                                       ; 24 cpi

; Addresses of command strings for each possible chars per inch
                .Word           0,0,0,0                                 ; 0-3
                .Word           PCT4CPI-PRCTable
                .Word           PCT5CPI-PRCTable
                .Word           PCT6CPI-PRCTable
                .Word           PCT7CPI-PRCTable
                .Word           PCT8CPI-PRCTable
                .Word           PCT9CPI-PRCTable
                .Word           PCT10CPI-PRCTable
                .Word           PCT11CPI-PRCTable
                .Word           PCT12CPI-PRCTable
                .Word           PCT13CPI-PRCTable
                .Word           0
                .Word           PCT15CPI-PRCTable
                .Word           0
                .Word           PCT17CPI-PRCTable
                .Word           0,0
                .Word           PCT20CPI-PRCTable
                .Word           0,0,0
                .Word           PCT24CPI-PRCTable

;               For ALQ, strings for outputting various commands
                .Word           0                       ; Initialize Terry
                .Word           $101-PRCTable           ; BE
                .Word           $102-PRCTable           ; BE
                .Word           $103-PRCTable           ; PB
                .Word           $104-PRCTable           ; PE
                .Word           $104-PRCTable           ; MB                    ; 5
                .Word           $103-PRCTable           ; ME
                .Word           0                       ; Do it Qume's way
                .Word           0                       ; Do it Qume's way
                .Word           $109-PRCTable           ; 6LPI
                .Word           $110-PRCTable           ; 8LPI                  ; 10.
                .Word           $111-PRCTable           ; Reset
                .Word           $112-PRCTable           ; Init
                .Word           0                       ; All CPI commands
                .Word           $118-PRCTable           ; P1
                .Word           $118-PRCTable           ; P2 gets P1 15.


                .Word           PCTWidths

                .Byte           0FF                     ; Custom Underlining
                .Byte           True                    ; Fractional justification

$101            .Byte           2,ComEscape,"O"
$102            .Byte           2,ComEscape,"R"
$103            .Byte           2,ComEscape,"D"
$104            .Byte           2,ComEscape,"U"
$109            .Byte           4,ComEscape,"L","0","8"
$110            .Byte           4,ComEscape,"L","0","6"
$111            .Byte           3,ComEscape,ComReturn,"P"
$112            .Byte           18.,ComEscape,1A,"I"
                .Byte           ComEscape,"L","0","8"
                .Byte           ComEscape,"X"                           ; Need b4 %
                .Byte           ComEscape,"%",ComEscape,1F,13.
                .Byte           ComEscape,"F","6","6"
$118            .Byte           0B.,ComEscape,"X"
                .Byte           ComEscape,"$"
                .Byte           ComEscape,"E","0","6"
```

-- continued next page --

```
; Actual strings to print various characters per inch.
PCT4CPI         .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","3","0"
PCT5CPI         .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","2","4"
PCT6CPI         .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","2","0"
PCT7CPI         .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","1","7"
PCT8CPI         .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","1","5"
PCT9CPI         .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","1","2"
PCT10CPI        .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","1","2"
PCT11CPI        .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","1","1"
PCT12CPI        .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","1","0"
PCT13CPI        .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","0","9"
PCT15CPI        .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","0","8"
PCT17CPI        .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","0","7"
PCT20CPI        .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","0","6"
PCT24CPI        .Byte           8,ComEscape,"X"
                .Byte           ComEscape,"%",ComEscape,"E","0","5"


; Dots per inch for Apple Silentype.  Basically an 80 column printers.
PrDTable        .Block          25.

; Table of dots used by characters at pitches.  Table runs from 0 to 24.
                .Block          25.

; Commands for various chars per inch.  This print doesn't know any.
                .Block          50.

; For various modes.  Silentype ain't that smart.
                .Block          32.

                .Word           0                       ; Widths

                .Byte           0                       ; Underlining
                .Byte           0

-- End of Part 2

-- end of file.12
```

Printer Definitions Work Area

Part III of seg.pr source code.

```
; Dots per inch for Qume and Diablo 16xx compat.  (Printer #17)
PrETable        .Byte           0,120.,120.,0,120.,120.,120.,120.    ; 0-7
                .Byte           120.,120.,120.,120.,120.,120.,0
                .Byte           120.,0,120.,0,0,120.,0,0,0
                .Byte           120.

; 120ths per character for valid charactrers 0 to 24.
                .Byte           0,0,0,0,30.,24.,20.,17.              ; 0-7
                .Byte           15.,13.,12.,11.,10.,9,0,8            ; 8-15. cpi
                .Byte           0,7,0,0,6,0,0,0                      ; 16-23. cpi
                .Byte           5                                    ; 24 cpi

; Addresses of command strings for each possible chars per inch
                .Word           0,0,0,0                              ; 0-3
                .Word           PET4CPI-PrETable
                .Word           PET5CPI-PrETable
                .Word           PET6CPI-PrETable
                .Word           PET7CPI-PrETable
                .Word           PET8CPI-PrETable
                .Word           PET9CPI-PrETable
                .Word           PET10CPI-PrETable
                .Word           PET11CPI-PrETable
                .Word           PET12CPI-PrETable
                .Word           PET13CPI-PrETable
                .Word           0
                .Word           PET15CPI-PrETable
                .Word           0
                .Word           PET17CPI-PrETable
                .Word           0,0
                .Word           PET20CPI-PrETable
                .Word           0,0,0
                .Word           PET24CPI-PrETable

;               For Qume, strings for outputting various commands
                .Word           0       ; Initialize Terry
                .Word           0       ; BB
                .Word           0       ; BE
                .Word           $103-PrETable       ; PB
                .Word           $104-PrETable       ; PE
                .Word           $104-PrETable       ; MB 5
                .Word           $103-PrETable       ; ME
                .Word           0                   ; UB
                .Word           0                   ; UE
                .Word           $109-PrETable       ; 6LPI
                .Word           $110-PrETable       ; 8LPI  10.
                .Word           $111-PrETable       ; Reset
                .Word           $112-PrETable       ; Init
                .Word           0                   ; All CPI commands
                .Word           $118-PrETable       ; P1
                .Word           $118-PrETable       ; P2 gets P1 15.

                .Word           PCTWidths           ; DWP widths
                .Byte           OFF                 ; Custom for Qume
                .Byte           True                ; Fractional justification

$103            .Byte           2,ComEscape,"D"
$104            .Byte           2,ComEscape,"U"
$109            .Byte           4,ComEscape,"L","0","8"
$110            .Byte           4,ComEscape,"L","0","6"
$111            .Byte           3,ComEscape,1A,"I"
$112            .Byte           18.,ComEscape,"L","0","8"
                .Byte           ComEscape,0E
                .Byte           ComEscape,"E","1","0"
                .Byte           ComEscape,"F","6","4"
                .Byte           ComEscape,"C","0","0"
$118            .Byte           6,ComEscape,0E
                .Byte           ComEscape,"E","0","6"
```

```
; Actual strings to print various characters per inch.
PFT4CPI         .Byte           6,ComEscape,0F,ComEscape,"E","3","0"
PFT5CPI         .Byte           6,ComEscape,0F,ComEscape,"E","2","4"
PFT6CPI         .Byte           6,ComEscape,0F,ComEscape,"E","2","0"
PFT7CPI         .Byte           6,ComEscape,0F,ComEscape,"E","1","7"
PFT8CPI         .Byte           6,ComEscape,0F,ComEscape,"E","1","5"
PFT9CPI         .Byte           6,ComEscape,0F,ComEscape,"E","1","3"
PFT10CPI        .Byte           6,ComEscape,0F,ComEscape,"E","1","2"
PFT11CPI        .Byte           6,ComEscape,0F,ComEscape,"E","1","1"
PFT12CPI        .Byte           6,ComEscape,0F,ComEscape,"E","1","0"
PFT13CPI        .Byte           6,ComEscape,0F,ComEscape,"E","0","9"
PFT15CPI        .Byte           6,ComEscape,0F,ComEscape,"E","0","8"
PFT17CPI        .Byte           6,ComEscape,0F,ComEscape,"E","0","7"
PFT20CPI        .Byte           6,ComEscape,0F,ComEscape,"E","0","6"
PFT24CPI        .Byte           6,ComEscape,0F,ComEscape,"E","0","5"


; Dots per inch for Epson FX.  (Printer #6) has 12 cpi also
PrFTable        .Byte           0,120.,120.,0,0,80.,96.,0       ; 0-7.
                .Byte           136.,0,80.,0,96.,0,0,0          ;8-15.
                .Byte           0,136.,0,0,0,0,0,0              ;16.-23.
                .Byte           0

; Dots per character for valid charactrers 0 to 24.
                .Byte           0,12.,12.,0,0,16.,16.,0         ; 0-7.
                .Byte           16.,0,8,0,8,0,0,0               ;8-15.
                .Byte           0,8,0,0,0,0,0,0                 ;16-23.
                .Byte           0                               ; 24.

; Addresses of command strings for each possible chars per inch
                .Word           0,0,0,0,0                       ; 0-4
                .Word           PFT9CPI-PrFTable
                .Word           PFT6CPI-PrFTable        ; 6 double 12
                .Word           0
                .Word           PFT8CPI-PrFTable        ; Double widith compress
                .Word           0
                .Word           PFT10CPI-PrFTable
                .Word           0
                .Word           PFT12CPI-PrFTable
                .Word           0,0,0,0                 ; 13-16
                .Word           PFT17CPI-PrFTable       ; 17
                .Block          14.                     ; 18-24


;               For Epson FX, strings for outputting various commands
                .Word           0                       ; Initialize Terry
                .Word           $101-PrFtable           ; BB
                .Word           $102-PrFtable           ; BE
                .Word           $103-PrFtable           ; PB
                .Word           $104-PrFtable           ; PE
                .Word           $105-PrFtable           ; MB 5
                .Word           $104-PrFtable           ; ME
                .Word           $107-PrFtable           ; UB    ; Requires pain in ass
                .Word           $108-PrFtable           ; UE
                .Word           $109-PrFtable           ; 6LPI
                .Word           $110-PrFtable           ; 8LPI  ; 10
                .Word           $111-PrFtable           ; Reset
                .Word           $111-PrFtable           ; Init
                .Word           0                       ; All CPI commands
                .Word           $114-PrFTable
                .Word           $114-PrFTable

                .Word           PFTWidths               ; FX does prop.
                .Byte           1                       ; Underalls
                .Byte           0       ; Fractional justification
```

-- continued next page

-- continued previous page

```
$101            .Byte       2,ComEscape,"G"         ; turn on bold
$102            .Byte       2,ComEscape,"H"         ; turn off bold
$103            .Byte       3,ComEscape,"S","0"
$104            .Byte       2,ComEscape,"T"
$105            .Byte       3,ComEscape,"S","1"
$107            .Byte       3,ComEscape,"-","1"
$108            .Byte       3,ComEscape,"-","0"
$109            .Byte       2,ComEscape,"2"         ; 6 lpi
$110            .Byte       2,ComEscape,"0"         ; 8 lpi
$111            .Byte       4,ComEscape,"@",ComEscape,"="
$114            .Byte       3,ComEscape,"p","1"

; Actual strings to cause various chars per inch
PFT5CPI         .Byte       7,ComEscape,"p","0",12,ComEscape,"P",0E
PFT6CPI         .Byte       7,ComEscape,"p","0",12,ComEscape,"M",0E
PFT8CPI         .Byte       7,ComEscape,"p","0",ComEscape,"P",0F,0E
PFT10CPI        .Byte       6,ComEscape,"p","0",12,ComEscape,"P"
PFT12CPI        .Byte       6,ComEscape,"p","0",12,ComEscape,"M"
PFT17CPI        .Byte       6,ComEscape,"p","0",ComEscape,"P",0F


; Dots per inch for Epson MX.  (Printer #7)
PrGTable        .Byte       0,0.,0.,0,0,80.,0,0     ; 0-7
                .Byte       134.,0,80.,0,0,0,0,0    ; 8-15.
                .Byte       0,136.,0,0,0,0,0,0      ; 16-23
                .Byte       0

; Dots per character for valid charactrers 0 to 24.
                .Byte       0,0,0,0,0,16.,0,0       ; 0-7
                .Byte       16.,0,8,0,0,0,0,0       ; 8-15.
                .Byte       0,8,0,0,0,0,0,0         ; 16-23.
                .Byte       0                       ; 24.

; Addresses of command strings for each possible chars per inch
                .Word       0,0,0,0,0               ; 0-4
                .Word       PGT5CPI-PrGTable
                .Word       0,0                     ; 6-7
                .Word       PGT8CPI-PrGTable        ; Double widith compress
                .Word       0
                .Word       PGT10CPI-PrGTable
                .Word       0,0,0,0,0,0             ; 11-16
                .Word       PGT17CPI-PrGTable       ; 17
                .Block      14.                     ; 18-24

;               For MX , strings for outputting various commands
                .Word       0                       ; Initialize Terry
                .Word       $101-PrGTable           ; BB
                .Word       $102-PrGTable           ; BE
                .Word       0           ; PB        ; Sub, super require Graftrax
                .Word       0           ; PE
                .Word       0           ; MB 5
                .Word       0           ; ME
                .Word       0           ; UB        ; Requires pain in ass
                .Word       0           ; UE
                .Word       $109-PrGTable           ; 6LPI
                .Word       $110-PrGTable           ; 8LPI  ; 10
                .Word       0           ; Reset
                .Word       0           ; Init
                .Word       0           ; All CPI commands
                .Word       0           ; not on hw
                .Word       0           ; not on HW

                .Word       0           ; No prop for MX
                .Byte       3           ; Underalls
                .Byte       0           ; Just

$101            .Byte       2,ComEscape,"G"  ; turn on bold
$102            .Byte       2,ComEscape,"H"  ; turn off bold
$109            .Byte       2,ComEscape,"2"  ; 6 lpi
$110            .Byte       2,ComEscape,"0"  ; 8 lpi
```

-- continued next page

-- continued previous page

```
; Actual strings to cause various chars per inch
PGT5CPI         .Byte           2,12,0E
PGT8CPI         .Byte           2,0F,0E
PGT10CPI        .Byte           2,12,14
PGT17CPI        .Byte           2,14,0F

; Dots per inch for Epson MX with Graftrax.  No 12 cpi
PrHTable        .Byte           0,0.,0.,0,0.,80.,0,0            ; 0-7
                .Byte           136.,0,80.,0,0,0,0,0            ; 8-15.
                .Byte           0,136.,0,0,0,0,0,0              ;16.-23.
                .Byte           0

; Dots per character for valid charactrers 0 to 24.
                .Byte           0,0,0,0,0,16.,0,0               ; 0-7
                .Byte           16.,0,8,0,0,0,0,0               ; 8-15.
                .Byte           0,8,0,0,0,0,0,0                 ;16-23.
                .Byte           0                               ; 24.

; Addresses of command strings for each possible chars per inch
                .Word           0,0,0,0,0                       ; 0-4
                .Word           PHT5CPI-PrHTable
                .Word           0,0                             ; 6-7
                .Word           PHT8CPI-PrHTable    ; Double width compress
                .Word           0                               ; 9
                .Word           PHT10CPI-PrHTable
                .Block          12.                             ; 11-16
                .Word           PHT17CPI-PrHTable               ; 17
                .Block          14.                             ; 18-24


;               For Epson MX Graftrax, strings for outputting various commands
                .Word           0                       ; Initialize Terry
                .Word           $101-PrHTable           ; BB
                .Word           $102-PrHTable           ; BE
                .Word           $103-PrHTable           ; PB
                .Word           $104-PrHTable           ; PE
                .Word           $105-PrHTable           ; MB 5
                .Word           $104-PrHTable           ; ME
                .Word           $107-PrHTable           ; UB
                .Word           $108-PrHTable           ; UE
                .Word           $109-PrHTable           ; 6LPI
                .Word           $110-PrHTable           ; 8LPI    ; 10
                .Word           $111-PrHTable           ; Reset
                .Word           $111-PrHTable           ; Init
                .Word           0       ; All CPI commands
                .Word           0       ; not on hw
                .Word           0       ; not on HW

                .Word           0                       ; No prop for MX graftrax
                .Byte           1
                .Byte           0                       ; Just

$101            .Byte           2,ComEscape,"G"                 ; turn on bold
$102            .Byte           2,ComEscape,"H"                 ; turn off bold
$103            .Byte           3,ComEscape,"S",0
$104            .Byte           4,ComEscape,"T",ComEscape,"H"
$105            .Byte           3,ComEscape,"S","1"
$107            .Byte           3,ComEscape,"-","1"
$108            .Byte           3,ComEscape,"-",0
$109            .Byte           2,ComEscape,"2"                 ; 6 lpi
$110            .Byte           2,ComEscape,"0"                 ; 8 lpi
$111            .Byte           2,ComEscape,"@"

; Actual strings to cause various chars per inch
PHT5CPI         .Byte           2,12,0E
PHT8CPI         .Byte           2,0F,0E
PHT10CPI        .Byte           2,12,14
PHT17CPI        .Byte           2,14,0F

; Dots per inch for Epson RX.  Has 12 cpi also.  Same as FX-proportional
PrITable        .Byte           0,0.,0.,0,0,80.,96.,0           ; 0-7
                .Byte           136.,0,80.,0,96.,0,0,0          ; 8-15.
                .Byte           0,136.,0,0,0,0,0,0              ;16.-23.
                .Byte           0
```

```
; Dots per character for valid characterers 0 to 24.
                .Byte           0,0,0,0,0,16.,16.,0             ; 0-7
                .Byte           16.,0,8,0,8,0,0,0               ; 8-15.
                .Byte           0,8,0,0,0,0,0,0                 ; 16-23.
                .Byte           0                               ; 24.

; Addresses of command strings for each possible chars per inch
                .Word           0,0,0,0,0                       ; 0-4
                .Word           PIT5CPI-PrITable
                .Word           PIT6CPI-PrITable    ; 6 CPI-PrITable is double 12
                .Word           0
                .Word           PIT8CPI-PrITable    ; Double width compress
                .Word           0
                .Word           PIT10CPI-PrITable
                .Word           0
                .Word           PIT12CPI-PrITable
                .Block          8                               ; 13-16
                .Word           PIT17CPI-PrITable               ; 17
                .Block          14.                             ; 18-24

;               For Epson RX, strings for outputting various commands
                .Word           0                       ; Initialize Terry
                .Word           $101-PrITable           ; BB
                .Word           $102-PrITable           ; BE
                .Word           $103-PrITable           ; PB
                .Word           $104-PrITable           ; PE
                .Word           $105-PrITable           ; MB 5
                .Word           $104-PrITable           ; ME
                .Word           $107-PrITable           ; UB    ; Requires pain in
ass
                .Word           $108-PrITable           ; UE
                .Word           $109-PrITable           ; 6LPI
                .Word           $110-PrITable           ; 8LPI  ; 10
                .Word           $111-PrITable           ; Reset
                .Word           $111-PrITable           ; Init
                .Word           0       ; All CPI commands
                .Word           0       ; not on hw
                .Word           0       ; not on HW

                .Word           0                       ; Not for RX
                .Byte           1       ; Underline
                .Byte           0       ; Just

$101            .Byte           2,ComEscape,"G"                 ; turn on bold
$102            .Byte           2,ComEscape,"H"                 ; turn off bold
$103            .Byte           3,ComEscape,"S","0"
$104            .Byte           2,ComEscape,"T"
$105            .Byte           3,ComEscape,"S","1"
$107            .Byte           3,ComEscape,"-","1"
$108            .Byte           3,ComEscape,"-","0"
$109            .Byte           2,ComEscape,"2"                 ; 6 lpi
$110            .Byte           2,ComEscape,"0"                 ; 8 lpi
$111            .Byte           2,ComEscape,"@"

; Actual strings to cause various chars per inch
PIT5CPI         .Byte           4,12,ComEscape,"P",0E
PIT6CPI         .Byte           4,12,ComEscape,"M",0E
PIT8CPI         .Byte           4,ComEscape,"P",0F,0E
PIT10CPI        .Byte           4,12,14,ComEscape,"P"
PIT12CPI        .Byte           4,12,14,ComEscape,"M"
PIT17CPI        .Byte           4,14,ComEscape,"P",0F


; Dots per inch for Custom printer
CustomTable     .Block          500.    ; Just leave room

-- End of Part 3

-- end of file.13
```

--- FILE.26 ---

Seq.pr Definition Module

```
;
; This book defines the area that comes in from seg.pr.  It is used in
; wp/print, db/rwseg, and ss/print, and iall/seg34.
;
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; WATCH OUT.  IF much is changed below, must change LPrFixed in iall/common.
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

; ALL THE FOLLOWING TABLES MUST BE IN SOME SORT OF ORDER, AS THEY
; ARE BULK MOVED INTO THE AREAS.

; Table of spacing intervals for various printers.
; Values are for 9, thru 17 cpi.  All values are based on non-proportional
; letters taking 8 increments.  Values 1 and 2 are for proportional 1 and 2.
; These are default values.  Can be overridden for specific printers.

; Dots per inch for 0-24. CPI
DPITable        .Equ            PrAreaStart     ; Set by caller

; Table of dots used by characters at pitches.  Table runs from 0 cpi
; to 24. cpi.
DPCTable        .Equ            DPITable+25.


; Table of words that point to commands that initiate various characters per
; inch.  If zero, not available on this printer.  Shouldn't happen.
CPIComms        .Equ            DPCTable+25.


; Addresses of strings that send the proper command to the selected printer.
; Value of zero means printer can't do it.  Value of <= FF is a Special
; routine number.  For control codes 00-15.  value of zero means it can't be
; done.
ControlCodes    .Equ            CPIComms+50.

CCEnd           .Equ            ControlCodes+32.

; Point at table of sizes, if any for this printer.
CSPointer       .Equ            CCEnd

; Which method is used for paper length commands.
ULMethod        .Equ            CSPointer+2     ; 1, 2 or 3
FracJust        .Equ            ULMethod+1      ; Boolean, can do it.

-- end of file.26
```

--- FILE.14 ---

How AppleWorks utilizes //gs memory manager

ALL //gs memory is managed by a portion of the toolkit called the Memory Manager, written by Rich Williams.

"Old style" programs, such as AppleWorks, must run at specific locations within the machine, such as most of the first 64K, and portions of the second 64K.

The memory manager originally starts with all of the RAM to manage, 256K on a basic gs, and 256K + 1024K on a gs with a 1 meg memory card.

Here's what happens to the memory:

During a cold boot, small portions of Ram, probably less than 32K, are requested by other toolkit modules.

Also during the cold boot, the RAMDisk is defined, but the actual memory allocated is based on the "minimum RamDisk size" that has been specified in the control panel. If this minimum is zero, then less than 10K bytes are actually allocated.

When AppleWorks boots, most of the first 64K of the gs, and portions of the second 64K, are reserved by AppleWorks for its use. This is done by making calls to the memory manager to reserve these specific areas.

At this point, very little of the RAM has been taken, and all that remains is considered, by AppleWorks, to be available for desktop. It is important to note that the RamDisk also thinks that this exact same unallocated Ram is available for extension of the RamDisk. This is one of the most powerful features of the gs toolkit.

The xxxK available stated by AppleWorks is the sum of:

K unallocated by the memory manager, plus
K that AppleWorks owns, minus
K that AppleWorks owns, but contains the files that are currently on the desktop.

As the user runs AppleWorks, and other programs, the unallocated Ram gets consumed by either AppleWorks, or the RamDisk, or other programs.

While AppleWorks is booting, it offers to preload the entire program. If the user allows this, AppleWorks acquires about 160K of Ram, from the memory manager, adding this memory to the pool owned by AppleWorks. Because of this feature, there are less reasons to copy the AppleWorks program files onto the RamDisk.

The AppleWorks owned pool consists of three types of Ram:

1. Files that appear on the desktop.
2. Portions, or all of, the AppleWorks program.
3. Free space

As in all earlier versions, AppleWorks will remove portions of itself from its pool if the space is needed to store data files.

There is a known bug in the RamDisk: If the RamDisk asks for a block of memory, from the memory manager, and none is available, the RamDisk will crash the whole system.

The solution for this bug, for AppleWorks users who also want to use the Ramdisk, is to have enough Ram so that the RamDisk can be initialized to a "minimum" that will almost always cover their potential use.

-- End of file.14

--- FILE.16 ---

Bugs and conflicts in AppleWorks version 2.0.

Thanks to the people that have reported these problems.

1. During startup, AppleWorks on a //gs sets the screen control byte at C035 to 1E, indicating that AppleWorks needs only the 80 column text screen.

   If a later desktop accessory subsequently wants to use the graphics screen, the gs memory manager has already allocated the graphics area for other uses, and can't make the space available for display.

   We do not have a recommended solution for the problem, and we would appreciate suggestions for changes to be made in AppleWorks to resolve the problem.

2. The screen control byte is set to 1E during startup. When AppleWorks quits it should be set back to its original setting, but this isn't done.

   Programs that follow AppleWorks without rebooting may find that the 1E setting conflicts with the programs need to do graphic screen display.

3. You can't enter Control-@ as a custom printer code. This bug may cause a change in AfReadTest in hooks (File 4).

4. Apple's //gs memory card is limited to 1 meg. Other vendors have subsequently gone much higher. AppleWorks 2.0 crashes if there is 4 meg or more. This will be fixed within 6 months, and we'll distribute a patch for the Startup disk if anyone actually buys 4 meg in the meantime.

5. There are no other known bugs in the memory management software. If you know of one, we would sure like to hear about it.

-- End of file.16

--- FILE.21 ---

LAN 2.1 Main Work Area

```
; Equates and definitions used throughout AppleWorks.

; This file is included in each AppleWorks segment.  Other developers may be
; interested in the individual components.  $600 byte work area defined here.

; Parameters that control assembly.  Combined refers to whether the segments
; are combined into one file.  They are for distribution, are not for my
; testing.

Combined        .Equ            1               ; 1 or 0, boolean
Crippled        .Equ            0
Language        .Equ            "A"
LAN             .Equ            0
Testing         .Equ            0
WarnEm          .Equ            0               ; 1 or 0, warn msg on splash

OAKey           .Equ            80              ; Open apple adds this
DateCode        .Equ            192.

;               Environment record information
ERELength       .Equ            36.             ; Length of ER Printer entry
EROFormFeed     .Equ            80
EROPageStop     .Equ            40
EroNeedLF       .Equ            20

False           .Equ            0

;               Bits to tell WriteVBar (VBarWrRtn) to write relative
;               to top left of box
FCLeftSide      .Equ            80
FCTopLine       .Equ            80

Indent          .Equ            80              ; Flags Indent needed
LPrFixed        .Equ            136.            ; Fixed area in printer defns.
MaxOnDesk       .Equ            12.
OldCode         .Equ            207.
PosSegs         .Equ            43.
PosSegsX4       .Equ            172.
ScreenWidth     .Equ            80.
TimeCode        .Equ            212.
True            .Equ            1
TurnOnAStack    .Equ            0C009
TurnOfAStack    .Equ            0C008
ZP              .Equ            0               ; For Mac conversion

; Items that control size of work areas, depending on machine size.
SmDBRecs        .Equ            1350.
SmWPRecs        .Equ            2250.
SmSSBA          .Equ            1550.

LrgDBRecs       .Equ            6350.
LrgWPRecs       .Equ            7250.
LrgSSBA         .Equ            10000.
```

```
; Items that control assembly for foreign languages.

KCommaInNo      .Equ            ","
KDateSep        .Equ            "/"
KDecPoint       .Equ            "."
KCurrency       .Equ            "$"
KVertBar        .Equ            "|"
No              .Equ            "N"
Yes             .Equ            "Y"
ComAlt1Help     .Equ            "/"+0AKey
ComAlt2Help     .Equ            "/"+0AKey
ComAlt3Help     .Equ            "/"+0AKey
ComClr2End      .Equ            25.             ; Control-Y
ComEditMode     .Equ            "E"+0AKey
ComEndSC        .Equ            "^"             ; To end printer special codes
ComHelp         .Equ            "?"+0AKey
ComHardCopy     .Equ            "H"+0AKey
ComQuick        .Equ            "Q"+0AKey
ComSave         .Equ            "S"+0AKey
LastLCChar      .Equ            "z"
LastUCChar      .Equ            "Z"
StdPL           .Equ            110.            ; Standard paper length
KSFMarker       .Equ            "^"             ; To display WP special functions.

; Hor and vertical for console driver.  Used in SubHost too.
CH              .Equ            14      ; Position rel to top left
CV              .Equ            15      ; Position rel to top left
KOAChar         .Equ            81
KMouseReturn    .Equ            8D

;               First the items that are common between all files
FileBuf         .Equ            0B00
HostWA          .Equ            0A00
LHostWA         .Equ            600             ; Length
HostLocn        .Equ            1000
LSane           .Equ            2100            ; Apple SANE numerics package
LElem           .Equ            0D000           ; Apple elementary functions
LHiSane         .Equ            21
LHiElems        .Equ            0D0
LZPSane         .Equ            0CC             ; Beg of 34 zero page locns.
LHiElem         .Equ            0D0
LSubHost        .Equ            0D000

DDWA            .Equ            7500            ; For disk directory work
SSWA            .Equ            7800
DBWA            .Equ            7B00
WPWA            .Equ            7B00

; Equates for relative addresses of main modules
LOrgMain        .Equ            2100
LSSMain         .Equ            6F00
LDBMain         .Equ            6B00
LWPMain         .Equ            6E00
LHiOrgMain      .Equ            21
LHiSSMain       .Equ            6F
LHiDBMain       .Equ            6B
LHiWPMain       .Equ            6E

; Equates for relative addresses of segments for each system.
LOrgSegs        .Equ            4000    ; 1.3 changed from 6300
LSSSegs         .Equ            3900
LDBSegs         .Equ            3900
LWPSegs         .Equ            2100

LSSOverlay      .Equ            5200

LHiOrgSegs      .Equ            40      ; 1.3 changed from 63
LHiSSSegs       .Equ            39
LHiDBSegs       .Equ            39
LHiWPSegs       .Equ            21

LHiSSOverlay    .Equ            52

IIMainBase      .Equ            8F00            ; First storage avail in main
IIHiMainBase    .Equ            8F
IIAuxBase       .Equ            0B00            ; First storage avail in aux.
IIHiAuxBase     .Equ            8
```

```
; The pathnames are new locations for 2.1 and 2.1 LAN
PathArea        .Equ            0BA00
MainPathName    .Equ            PathArea        ; String
;               If HardPath is 0, then use this Pathname.
;               If Hard <> 0, then this is just for display.
UFPathName      .Equ            MainPathName+50.        ; User folder Pathname
CodePathName    .Equ            UFPathName+50.
ERMainPathName  .Equ            CodePathName+50.

;               For //e bank addressing.  Values temp room for SubHost.
IOBuffer        .Equ            0BB00

PDDevCntM1      .Equ            0BF31           ; ProDOS devices attached
PDDevLst        .Equ            0BF32           ; ProDOS devices attached

MaxSSRows       .Equ            999.
RdMain48        .EQU            0C002
RdAux48         .EQU            0C003
WrMain48        .EQU            0C004
WrAux48         .EQU            0C005
RWAux16         .EQU            0C009
RWMain16        .EQU            0C008

; Type codes for numeric package
FFExt           .Equ            0
FFDbl           .Equ            1
FFInt           .Equ            4

; Op codes for numerics package
FOAdd           .Equ            0
FOSetEnv        .Equ            1
FOSub           .Equ            2
FOGetEnv        .Equ            3
FOMult          .Equ            4
FODiv           .Equ            6

FOD2B           .Equ            9
FOB2D           .Equ            0B

FOSQRT          .Equ            12
FORTI           .Equ            14      ; Round to integer
FOTTI           .Equ            16      ; Truncate to integer
FOClass         .Equ            1C
FOABS           .Equ            0F      ; Absolute
FOCMP           .Equ            08      ; Compare
FONeg           .Equ            0D
FOZ2X           .Equ            0E
FOX2Z           .Equ            10
LExtNum         .Equ            10.

; And opcodes for Elems package
FOXPwrY         .Equ            12

; Application call codes.  Host will place these values in X when making
; calls to the individual Main routines.  A high bit added means to just get
; the proper modules loaded for this, but not called.
;                                 Returns to
ACCloseFile     .Equ            1   ; Caller, put back in banke, release WA.
ACInitFile      .Equ            2   ; Caller, is in WA, needs pointers inited.
ACMergeFile     .Equ            80  ; Host.
ACNewFrKB       .Equ            84  ; Host.
ACNewFrCB       .Equ            85  ; Host, Clipboard.
ACOpenFile      .Equ            6   ; Caller, Gets from banks, does init.
ACReleaseFile   .Equ            7   ; Caller, release the bank switched.
ACResumeFile    .Equ            88  ; Host, already in work area, init done.

IOFailed        .Equ    ZP+003
```

-- continued next page

-- continued previous page

```
;               All of the following depend on whether or not Slinky
;               is working.
ALVDBMMP        .Equ    ZP+004  ; A(Ptr for maximum record)
ALVWPMMP        .Equ    ZP+006  ; A(Ptr for maximum line)
AEndSSWA        .Equ    ZP+008  ; A(Last byte of BldRecWA)
MaxDBRecs       .Equ    ZP+00A  ; Maximum recs allowed in DB
MaxWPLines      .Equ    ZP+00C  ; Maximum lines allowed in WP
MaxSSRSize      .Equ    ZP+00E  ; Maximum record size in SS

AuxReg          .Equ    ZP+080
CDAddr          .Equ    ZP+080          ; Used by console driver
CurHor          .Equ    ZP+082          ; Cursor position
CurVer          .Equ    ZP+083          ; Cursor position 0..21
CWork1          .Equ    ZP+084
SpaceOK         .Equ    ZP+085          ; Cleared, set by SMPut routines.
FoundCommand    .Equ    ZP+086
FoundReturn     .Equ    ZP+087
FoundEscape     .Equ    ZP+088          ; Non zero if escape found
HelpAvail       .Equ    ZP+089
I               .Equ    ZP+08A
IntFrKB         .Equ    ZP+08C          ; numeric
LLR             .Equ    ZP+08D
LLB             .Equ    ZP+08F          ; is a word

; Work area for Multiply
MReg            .Equ    ZP+091
MWrkX           .Equ    ZP+093
MWrkY           .Equ    ZP+094
PageBegin       .Equ    ZP+095
PageEnd         .Equ    ZP+096
ReturnCode      .Equ    ZP+097

; Return 0 through ZReg02 must be in order
RETURN0         .EQU    ZP+098
RETURN2         .EQU    ZP+09A
AArg            .Equ    ZP+09A
RETURN4         .EQU    ZP+09C
BArg            .Equ    ZP+09C
ZReg00          .Equ    ZP+09E
CArg            .Equ    ZP+09E
ZReg02          .Equ    ZP+0A0
DArg            .Equ    ZP+0A0
R1              .Equ    ZP+0A2
ExitFlag        .Equ    ZP+0A4  ; HAS DA-D or DA-S that caused it
                                ; to bail out of the current file
Work1           .Equ    ZP+0A5
FR              .EQU    ZP+0A6
FBR             .Equ    ZP+0D0
;

; These are output to the screen codes; For My ConsDriver.
ClearELine      .Equ    1
ClearLine       .Equ    2
ClearVP         .Equ    3
ClearEVP        .Equ    4
AbsPosn         .Equ    5
Left            .Equ    6
Right           .Equ    7
Up              .Equ    8
Down            .Equ    9
Inverse         .Equ    10.
Normal          .Equ    11.
VPBot           .Equ    12.
Return          .Equ    13.
VPTop           .Equ    14.
VPReset         .Equ    15.
ResetVP         .Equ    15.
Bell            .Equ    16.
HorShiftVP      .Equ    17.
```

-- continued next page

-- continued previous page

```
; These are input codes, some output too.
ComCurLeft      .EQU    8
ComCurRight     .EQU    21.
ComCurUp        .EQU    11.
ComCurDown      .Equ    10.
ComTabLeft      .Equ    137.
ComTabRight     .Equ    9.
ComEscape       .EQU    27.
ComDelKey       .Equ    7F
ComReturn       .Equ    13.

ComLineFeed     .Equ    10.

ComFormFeed     .Equ    0C


Buffer          .Equ            HostWA

;               All the stuff for the clipboard.  This must be in order
;               as it is cleared as one block of 515 bytes.
KlipBoard       .Equ            Buffer+80.

; KBHigest is not consistant between applications:
; WP:  The count of items on the clipboard
; DB:  The highest place used, values from 0 - 255. (0: 1 place used)
KBHighest       .Equ            KlipBoard+512.
KBType          .Equ            KBHighest+2     ; Type of data L C T or M
;               End of what must be in order

CurrFNo         .Equ            KBType+1

;               Stuff for organizer, 12 items on desk.
DTCurOpen       .Equ            CurrFNo+1
DTCtOnDesk      .Equ            DTCurOpen+1

;               The file that is actually open on the desk, 32 bytes
DTFName         .Equ            DTCtOnDesk+1
DTFType         .Equ            DTFName+21.
DTFStatus       .Equ            DTFType+1
DTFPicked       .Equ            DTFStatus+1     ; Boolean:  Picked for save
DTFSizeK        .Equ            DTFPicked+1     ; 1.3 changed to 2 bytes
DTFAddress      .Equ            DTFSizeK+2      ; Bank address if not curr.


;               Pointers to the bank switched places where the up to
;               12 files are stored.  When one is removed, all higher
;               are moved down so that the n that are in, are contiguous.
;               Leave the first unused, so that accum will point right
DTFPointers     .Equ            DTFName+32.

FieldName       .Equ            DTFPointers+26.
FoundSpace      .Equ            FieldName+21.
HardPathName    .Equ            FoundSpace+1
                                ;  Prodos DSSS0000 device number.  If non
                                ; zero, use this to find a file catalog
                                ; SOS value 01 to 06, locn of file cat.
; Lost4Ever used to be MainPathName.  Suggested convention:  Leave
; first two bytes zero.  Remaining 30 are completely available
; as far as AppleWorks is concerned.
Lost4Ever       .Equ            HardPathName+1

; Following is new for 2.1 and 2.1 LAN
UserID          .Equ    Lost4Ever+2     ; string max len=8 (ie, 9 bytes)
LaunchLev       .Equ    UserID+9        ; One byte
```

-- continued next page

-- continued previous page

```
; Where we keep the stack of routine names
CurrMode        .Equ            Lost4Ever+32.
StackP1         .Equ            CurrMode+21.
StackP2         .Equ            StackP1+21.
StackP8         .Equ            StackP2+126.

NewFile         .Equ            StackP8+21.
Newstr          .Equ            NewFile+1
OldStr          .Equ            NewStr+128.     ; Max length 127.
OneChar         .Equ            OldStr+128.
OrgInMem        .Equ            OneChar+1       ; Organizer in memory
RacPDNo         .Equ            OrgInMem+1
R2              .Equ            RacPDNo+1
R3              .Equ            R2+2

SavedHor        .Equ            R3+2            ; MUST be Hor, then ver
SavedVer        .Equ            SavedHor+1
SegNum          .Equ            SavedVer+1      ; SOS only segment number
SMAddr0         .Equ            SegNum+1        ;
WVVAddr         .Equ            SMAddr0+2
WVVCount        .Equ            WVVAddr+2
VertTable       .Equ            WVVCount+2      ; X, Y, Length, entries 0-30
VTHighest       .Equ            VertTable+93.   ; Highest entry in table
Work2           .Equ            VTHighest+1
Work3           .Equ            Work2+2

; Environment record has starting file cabinet, and printer specifications
; IF ANY OF THIS CHANGES, YOU HAVE TO CHANGE SEGPR, WHICH CONTAINS
; THE INITIAL VALUES FOR ALL OF THIS.
EnvRec                  .Equ            Work3+3
ERHardPathName          .Equ            EnvRec
ERAvail                 .Equ            ERHardPathName+1 ; Avail 2.1
ERPtrCount              .Equ            ERAvail+32.             ; 0-3
EROAWPrinter            .Equ            ERPtrCount+1            ; 0-3
ERMainPrinter           .Equ            EROAWPrinter+1          ; 0-3
ERCurrPrinter           .Equ            ERMainPrinter+1
ERToday                 .Equ            ERCurrPrinter+1
ERFiller1               .Equ            ERToday+9 ; String Normally [I] 80N

; The following is repeated three times
ERPtrName               .Equ            ERFiller1+10.
ERDriverName            .Equ            ERPtrName+16.   ; Nada on //e, codes
                                                        ; to init port on //c
; Next says spec rules for //c (c), //e (e), or on disk (d)
ERSpecOn                .Equ            ERDriverName+15. ; Where custom made.
ERDeviceNo              .Equ            ERSpecOn+1      ; Or slot or FF (disk)
ERPtrType               .Equ            ERDeviceNo+1
EROptions               .Equ            ERPtrType+1     ; Boolean
ERPlaten                .Equ            EROptions+1

EnvRecEnd               .Equ            ERPtrName+108.
; End of part of environment record in memory.

; Amount to move into memory at startup
EnvRecALength           .Equ            EnvRecEnd-EnvRec
; So other segments can get past the environemtn record on disk
; 50 added for new position of ERMainPathName (after third printer)
EnvRecRLength           .Equ            EnvRecALength+50.
D100Length              .Equ            300     ; Space saved in Seg.pr

; Variables that specify current (Organizer) file card, and position
; of its top left corner
DFCNumber       .Equ    EnvRecEnd
DFCHorPosn      .Equ    DFCNumber+1     ; Hor posn of left side
                                        ; Includes preceding 2 sp.
DFCVerPosn      .Equ    DFCHorPosn+1    ; Vert posn of top left byte

;               Boolean whether code is coming from Profile
CodeOnFive      .Equ    DFCVerPosn+1    ; Initial is zero (false)
ClockByte       .Equ    CodeOnFive+1    ; Byte at 0BF06 (ProDOS clock)
CSWValue        .Equ    ClockByte+1     ; 2 bytes, value of 36,37 at entry
Clearing        .Equ    CSWValue+2      ; Boolean, ReadKB is for ClearTA

PrSlots         .Equ    Clearing+1      ; //e, slots 0-7 valid for pr, bool
DidSwapDisk     .Equ    PrSlots+8       ; Boolean, data disk in drive 1
```

-- continued next page

```
-- continued previous page

FloppyList      .Equ    DidSwapDisk+1   ; list of drives, like 60,E0.
                                        ; For SOS, first byte is highest drive #,
                                        ; like X'02', and remainder is not used

LastDBPrinter   .Equ    FloppyList+6
LastWPPrinter   .Equ    LastDBPrinter+1
LastSSPrinter   .Equ    LastWPPrinter+1

Today           .Equ    LastSSPrinter+1
SlinkAddr       .Equ    Today+20.       ; Value for X reg., from BFF8
SlinkSlot       .Equ    SlinkAddr+1

ConsDevNum      .Equ    SlinkSlot+1     ; PROBABLY AVAILABLE.
IsLolly         .Equ    ConsDevNum+1

; RamDisk driver address and device number
RamDiskD        .Equ    IsLolly+1
RamDiskN        .Equ    RamDiskD+2

; Routine for when desktop is full
DontBitch       .Equ    RamDiskN+1

PEChar          .Equ    DontBitch+1     ; Char for bottom line.
KDTAvail        .Equ    PEChar+1        ; 1.3 Word: Last K bytes available.
SlinkMin        .Equ    KDTAvail+2      ; 1.3 Size of slinky blocks.
SlinkShifts     .Equ    SlinkMin+1      ; 1.3 How many times to rotate on Slinky
StrWork5        .Equ    SlinkShifts+1   ; 1.3 6 bytes from Word2Str
KBWidth         .Equ    StrWork5+6      ; 55 only so far.
MyID            .Equ    KBWidth+1       ; 2 bytes, Cortland
Seg00Type       .Equ    MyID+2          ; 1 byte, which seg00 loaded

; Typeahead for Apple //e
TAArea          .Equ    Seg00Type+1
TALength        .Equ    24.
TANextIn        .Equ    TAArea+TALength
TANextOut       .Equ    TANextIn+1

end of file.22
```

--- FILE.22 ---

LAN 2.1 Host Entry Points

```
; These are references to the routines in Host
; Skip          JMP     Relocate
; Skip          .Byte   Version

ABuffer         .Equ    HostLocn+4
ACallSegTab     .Equ    ABuffer+2

B4ReadTest      .Equ    ACallSegTab+2
AfReadTest      .Equ    B4ReadTest+4
B4WriteTest     .Equ    AfReadTest+4
B4ElemsTest     .Equ    B4WriteTest+4

; Variables common to all routines
StrWork3        .Equ    B4ElemsTest+4
MainInMem       .Equ    StrWork3+4
TaskTable       .Equ    MainInMem+1
PrevSegLoad     .Equ    TaskTable+4

CallSegTab      .Equ    PrevSegLoad+1

CByte           .Equ    CallSegTab+PosSegsX4+4
Available       .Equ    CByte+1

StrikeOver      .Equ    Available+30.
CursorOnSw      .Equ    StrikeOver+1
LeaveAlone      .Equ    CursorOnSw+1
ChUnderCur      .Equ    LeaveAlone+2
SaveE1          .Equ    ChUnderCur+1

AskForProg      .Equ    SaveE1+1
CallSeg         .Equ    AskForProg+3
CC2S            .Equ    CallSeg+3
ClearDA         .Equ    CC2S+3
ClearTA         .Equ    ClearDA+3
ClearWindow     .Equ    ClearTA+3
ConsDriver      .Equ    ClearWindow+3
Conv1TP         .Equ    ConsDriver+3
DieNow          .Equ    Conv1TP+3
DivWord         .Equ    DieNow+3
DoBell          .Equ    DivWord+3
DoGoToXY        .Equ    DoBell+3
DoHelp          .Equ    DoGoToXY+3
DTTestAvail     .Equ    DoHelp+3
DTIsFull        .Equ    DTTestAvail+3
EnterCommand    .Equ    DTIsFull+3
FullExit        .Equ    EnterCommand+3  ; Z is to see if called anywhere
GetScrLine      .Equ    FullExit+3
GetArgs         .Equ    GetScrLine+3
GetVBar         .Equ    GetArgs+3
HighLight       .Equ    GetVBar+3
```

```
IODisable       .Equ    HighLight+3
IOEnable        .Equ    IODisable+3
MakeIB          .Equ    IOEnable+3

MadeAChange     .Equ    MakeIB+3
MultByte        .Equ    MadeAChange+3
MultWord        .Equ    MultByte+3
MvLeftRtn       .Equ    MultWord+3
MvRightRtn      .Equ    MvLeftRtn+3
PopStack        .Equ    MvRightRtn+3
PressAny        .Equ    PopStack+3
Print           .Equ    PressAny+3
PushStack       .Equ    Print+3
ReadKB          .Equ    PushStack+3
RestCursor      .Equ    ReadKB+3
SaveGoToXY      .Equ    RestCursor+3
SetUCC          .Equ    SaveGoToXY+3
SetUCS          .Equ    SetUCC+3
StrCmpRtn       .Equ    SetUCS+3
StrMvRtn        .Equ    StrCmpRtn+3
StrWrRtn        .Equ    StrMvRtn+3
Trap            .Equ    StrWrRtn+3
VBarWrRtn       .Equ    Trap+3
Wait            .Equ    VBarWrRtn+3
WipeOutTA       .Equ    Wait+3
Write           .Equ    WipeOutTA+3
Writ1Byte       .Equ    Write+3
WritePE         .Equ    Writ1Byte+3
WritPRtn        .Equ    WritePE+3
WriteCom        .Equ    WritPRtn+3
WriteRUSure     .Equ    WriteCom+3

-- end of file.22
```

--- FILE.23 ---

LAN 2.1 Sub-host Entry Points

```
; These are references to the routines in SubHost
SMAvailCount    .Equ    LSubHost+2              ; 1.3 name changed

SMGetBlock      .Equ    SMAvailCount+3
SMGetCell       .Equ    SMGetBlock+3
SMGetStrings    .Equ    SMGetCell+3
SMMaxCol        .Equ    SMGetStrings+3
SMPutBlock      .Equ    SMMaxCol+3
SMPutCell       .Equ    SMPutBlock+3
SMPutStrings    .Equ    SMPutCell+3
SMRelBlock      .Equ    SMPutStrings+3
SMRetBSize      .Equ    SMRelBlock+3            ; 1.3 Name changed
SMRetWPSize     .Equ    SMRetBSize+3            ; 1.3 Name changed

Conv1TDec       .Equ            SMRetWPSize+3
DispEsc         .Equ            Conv1TDec+3
DispFR          .Equ            DispEsc+3
DispFE          .Equ            DispFR+3
GetDec          .Equ            DispFE+3
GetMenuBar      .Equ            GetDec+3
GetNum          .Equ            GetMenuBar+3
GetStr          .Equ            GetNum+3
GetYN           .Equ            Getstr+3
PDSN            .Equ            GetYN+3
ReleaseKB       .Equ            PDSN+3
Word2Str        .Equ            ReleaseKB+3     ; 1.3 name changed.
UndoHelp        .Equ            Word2Str+3
VerifyDelete    .Equ            UndoHelp+3
FirstEmpty      .Equ            VerifyDelete+3  ; Two bytes
SetAFP          .Equ            FirstEmpty+2
```

-- end of file.23

--- FILE.31 ---

LAN 2.1 Data Base Common Work Area

```
; Equates and zero page locations for Data Base

Task            .Equ        "F"             ; ie., Data Base
MainSegNo       .Equ        1               ; Main is seg.01
MaxFields       .Equ        30.
MaxRpWidth      .Equ        180.
RptSizeH        .Equ        620.            ; Bytes for tables style
RptSizeV        .Equ        450.            ; For labels style
VRLine1         .Equ        7
KAuxBase        .Equ        33      ; Rel addresses >= are in aux.

; For report formats (Tables style)
LRptArea        .Equ            600.
LCalcArea       .Equ            54.

; Zero page addresses

BR              .Equ    ZP+0A8              ; For file loading
CR              .Equ    ZP+0AA              ; Column register
HR              .Equ    ZP+0AE              ; Just in case
RR              .Equ    ZP+0B0

ComArrange      .Equ                "A"+0AKey
ComCopy         .Equ                "C"+0AKey
ComDitto        .Equ                "'"+0AKey
ComDelete       .Equ                "D"+0AKey
ComFind         .Equ                "F"+0AKey
ComGroup        .Equ                "G"+0AKey
ComInsert       .Equ                "I"+0AKey
ComJustify      .Equ                "J"+0AKey
ComCalcCol      .Equ                "K"+0AKey
ComLayout       .Equ                "L"+0AKey
ComMove         .Equ                "M"+0AKey
ComChName       .Equ                "N"+0AKey
ComPrOptions    .Equ                "O"+0AKey
ComPrint        .Equ                "P"+0AKey
ComRecSel       .Equ                "R"+0AKey
ComTotals       .Equ                "T"+0AKey
ComStandard     .Equ                "V"+0AKey
ComPrFN         .Equ                "V"+0AKey
ComZoom         .Equ                "Z"+0AKey
ComPAGEBACK     .Equ                11.+0AKey
ComPAGEFORW     .Equ                10.+0Akey

; The following is the layout of the Data base  work area.  The beginning
; address is somewhat variable until this settles down.
                .List

;                   First the items that are common between all files

; Begin the file header record
DBHeader        .Equ            DBWA
ABReports       .Equ            DBHeader        ; Address of record pointers
BABARecords     .Equ            ABReports+16.   ; Bank adr of block of addr.
                                                ; of records

; Now file records one and two, exactly
BasicMode       .Equ            DBWA+24.
CurRecNo        .Equ            BasicMode+1
CurrRpNo        .Equ            CurRecNo+2
FVSequence      .Equ            CurrRpNo+1
FHRetDir        .Equ            FVSequence+1
HDCLine         .Equ            FHRetDir+1
HDCColumn       .Equ            HDCLine+1
Mode            .Equ            HDCColumn+1
NFields         .Equ            Mode+1
NRecords        .Equ            NFields+1
NReports        .Equ            NRecords+2
PSL             .Equ            NReports+1
VDCField        .Equ            PSL+2
```

```
;                   Blocks of info for up to 30 fields, plus some slack
FHHor               .Equ                VDCField+1
FHField             .Equ                FHHor+36.
FVHor               .Equ                FHField+36.
FVVer               .Equ                FVHor+36.
FVField             .Equ                FVVer+36.
FHFields            .Equ                FVField+36.

; Select data from file.  Has to stay as shown in Pascal
SelectOn            .Equ                FHFields+1
SelectTest          .Equ                SelectOn+6
SelectCont          .Equ                SelectTest+6
SelectData          .Equ                SelectCont+6
Filler1             .Equ                SelectData+96.
FHNames             .Equ                Filler1+20.
; This is the end of the exact format of the input file

; The number of FHNames stored will be same as NFields; 22 bytes each.

;                   Work area to store record pointers.  1025 for //e,
;                   3073 for ///.  Record 0 is defaults, 1-n are real.

BldRecWA            .Equ                FHNames+660.
BldRecTop           .Equ                BldRecWA+1024.

; Information that is built while a file is open, and not important
; to save when closed


CRSPT               .Equ                BldRecTop
CurrRpName          .Equ                CRSPT+60.
ConvError           .Equ                CurrRpName+21.
ConvMore            .Equ                ConvError+1
DateIn              .Equ                ConvMore+1
DateGood            .Equ                DateIn+21.
XFILLER             .Equ                DateGood+1
FFSWitch            .Equ                XFILLER+1
FFData              .Equ                FFSWitch+1
IsDateF             .Equ                FFData+32.
IsTimeF             .Equ                IsDateF+1
L15Work             .Equ                IsTimeF+1
MaxLength           .Equ                L15Work+16.
Numstr              .Equ                MaxLength+1
NFieldsX2           .Equ                NumStr+6
NewStr1             .Equ                NFieldsX2+1
SaveCWork1          .Equ                Newstr1+1
SFDisplace          .Equ                SaveCWork1+1
SortL               .Equ                SFDisplace+1
SaveFR              .Equ                SortL+1
TopRecNo            .Equ                SaveFR+2
WorkPointer         .Equ                TopRecNo+2

; Following are not kept on file
SelectType          .Equ                WorkPointer+2
SelectNumeric       .Equ                SelectType+6

; Save selection rules while in del mode, or report
SaveRacSel          .Equ                SelectNumeric+48.
IFCEnd              .Equ                SaveRacSel+114.

; Area is 1350*2 or 6350*2, depending on whether Slinky present.
ARecords            .Equ                IFCEnd

                    .If                 ARecords+SmDBRecs+SmDBRecs+1>IIMainBase
                    Crappity
                    .Endc

                    .If                 ARecords+LrgDBRecs+LrgDBRecs+1>PathArea
                    Crappity
                    .Endc

-- end of file.31
```

--- FILE.38 ---

LAN 2.1 Common Work Area

```
; Common routines for SpreadSheet assembly
;
MainSegNo       .Equ    24.             ; Main for SS is seg.24
Task            .Equ    "C"

ComArrange      .Equ    "A"+0AKey
ComBlank        .Equ    "B"+0AKey
ComCopy         .Equ    "C"+0AKey
ComDelete       .Equ    "D"+0AKey
ComFind         .Equ    "F"+0AKey
ComInsert       .Equ    "I"+0AKey
ComJump         .Equ    "J"+0AKey
ComCalc         .Equ    "K"+0AKey
ComLayout       .Equ    "L"+0AKey
ComMove         .Equ    "M"+0AKey
ComChName       .Equ    "N"+0AKey
ComOptions      .Equ    "O"+0AKey
ComPrint        .Equ    "P"+0AKey
ComTitles       .Equ    "T"+0AKey
ComEdit         .Equ    "U"+0AKey
ComStandard     .Equ    "V"+0AKey
ComView         .Equ    "W"+0AKey
ComZoom         .Equ    "Z"+0AKey
KFormSep        .Equ    ","             ; Separates list items in formula

; Bits for WorkType byte +0
BitValue        .Equ    80
BitNDIfZero     .Equ    40              ; Values solamente
BitProp         .Equ    20              ; Just for labels
BitSimValue     .Equ    20              ; Just for values.
BitProNL        .Equ    10              ; Labels not allowed
BitProNV        .Equ    8               ; Values not allowed

BitsVFS         .Equ    1               ; Bits Value Format Standard
BitsVFF         .Equ    2               ; For values
BitsVFD         .Equ    3
BitsVFC         .Equ    4
BitsVFP         .Equ    5
BitsVFA         .Equ    6

BitsLFS         .Equ    1
BitsLFL         .Equ    2               ; For labels
BitsLFR         .Equ    3
BitsLFC         .Equ    4


; Bits for WorkType byte +1 (just for values)
BitCalced       .Equ    80
BitNA           .Equ    40
BitError        .Equ    20
BitMustCalc     .Equ    10              ; Must calc this time.

; These are internal codes found in internally stored strings
KIntCoor        .Equ            254.    ; Next two bytes are XY
KIntNum         .Equ            253.    ; Next six bytes are number
KIntDots        .Equ            252.    ; Replaces ...
KAUnPlus        .Equ            251.
KAUnMinus       .Equ            250.
KALParen        .Equ            249.
KAMult          .Equ            248.
KADiv           .Equ            247.
KAPlus          .Equ            246.
KAMinus         .Equ            245.
KARParen        .Equ            244.
KAExp           .Equ            243.
KAComma         .Equ            242.    ; Comma within function
```

-- continued next page

-- continued previous page

```
;                   Logical operators
KALessThan          .Equ                241.
KAGtrThan           .Equ                240.
KAEquals            .Equ                239.
KALessEqual         .Equ                238.
KAGtrEqual          .Equ                237.
KANotEqual          .Equ                236.

;                   This is just used by replicate
KAbsCoor            .Equ                210.

;                   Now start the functions
KFRound             .Equ                217.
KFOr                .Equ                218.
KFAnd               .Equ                219.
KFSum               .Equ                220.        ; First because most common
KFAvg               .Equ                221.
KFChoose            .Equ                222.
KFCount             .Equ                223.
KFError             .Equ                224.
KFIRR               .Equ                225.
KFIF                .Equ                226.
KFINT               .Equ                227.
KFLookup            .Equ                228.
KFMax               .Equ                229.
KFMin               .Equ                230.
KFNA                .Equ                231.
KFNPV               .Equ                232.
KFSQRT              .Equ                233.
KFAbs               .Equ                234.

;                   Now some constants
KHighCol            .Equ                127.
KFirstDLine         .Equ                2
KDispLines          .Equ                18.         ; 18 lines on screen
KLastDLine          .Equ                19.
LIntNum             .Equ                8           ; Length of internal numerics storage

; Zero page addresses for spreadsheet
CurRow              .Equ    ZP+0AB                  ; Current Row (1-999)
CurCol              .Equ    ZP+0AD                  ; Current Col (1-127)
WorkRow             .Equ    ZP+0AE                  ; Used by many routines
WorkCol             .Equ    ZP+0B0                  ; Used by many routines

; Provided by GetSpecs
CCWidth             .Equ    ZP+0B1                  ; Width of current column
HOnScreen           .Equ    ZP+0B2                  ; 0: no 1: partial, 2: fully
VOnScreen           .Equ    ZP+0B3                  ; 0: no, 1: Yes
MaxLength           .Equ    ZP+0B4

; Provided by AccumType
IsParen             .Equ    ZP+0B5
IsOpr               .Equ    ZP+0B6
IsAlpha             .Equ    ZP+0B7
IsNum               .Equ    ZP+0B8
IsBuiltIn           .Equ    ZP+0B9


SwDispBox           .Equ    ZP+0BA      ; Redisplay the entire page, including
MakeIB
SwDisp12            .Equ    ZP+0BB      ; Redisplay all rows within box
SwDispPl            .Equ    ZP+0BC      ; Redisplay line 23
SwDispCur           .Equ    ZP+0BD      ; Inverse the cursor

ValidMoves          .Equ    ZP+0BE      ; Valid cursor moves, see next two lines
ValidRows           .Equ    80
ValidCols           .Equ    40
ValidAny            .Equ    80+40
DispFRows           .Equ    20      ; Tell expand cursor to display full rows
DispFCols           .Equ    10      ; Tell expand cursor to display full cols
Valid1Dim           .Equ    08      ; Only allow expand in one direction

IsArrow             .Equ    ZP+0BF
SwRecalc            .Equ    ZP+0C0
```

```
; Zero page locations for convert to internal
CX2IVal1        .Equ            ZP+0C1
CX2IVal2        .Equ            ZP+0C2

; Bits for the above two bytes, indicating what's valid right now
OKUnPM          .Equ            80      ; Unury plus or minus
OKOpr           .Equ            40      ; * + - / or ^
OKDots          .Equ            20      ; ... (between operators
OKRParen        .Equ            10      ; Right paren
OKAt            .Equ            08      ; @ sign
OKDecPt         .Equ            04      ; Begins a number
OKLParen        .Equ            02      ; Left paren
OKAlpha         .Equ            01      ; A to Z, starting coordinates
OKNum           .Equ            80      ; Start CX2IVal2, numeric
OKArrow         .Equ            40      ; Can use arrows to find cell
OKComma         .Equ            20      ; Comma is OK
OKEnd           .Equ            10      ; Ok to end now, by CR or arrow.


; Zero page assignment:
;  0AB - 0C8    Used throughout, defined here
;  0C9 - 0D8    Defined and used in Main only
;  0D9 - 0E4    Defined and used by EditM
;  0E4 - 0EF    Defined and used by EditE, EditL, etc.

; The following is the layout of the spreadsheet work area.

; Information the file that must be kept while the file is in memory,
; but closed
FCFRows         .Equ            SSWA
FCFCols         .Equ            FCFRows+1000.   ; 128 bytes

; Try to keep SSHeader aligned where it was in 2.0
SSHeader        .Equ            SSWA+1280.      ; File header record
ColWidths       .Equ            SSHeader+3      ; 128 bytes

;                Global variables
GloOrder        .Equ            ColWidths+128.  ; Initial: Appropriate
GloReCalc       .Equ            GloOrder+1      ; Initial: Automatic
SvLocn          .Equ            GloReCalc+1 ; CurRow,CurCol

; Information that controls the views
VIViews         .Equ    SvLocn+3        ; Split "1 S(ide by side, T(op and
bottom
VISync          .Equ    VIViews+1       ; Boolean, sync or not if two views

; Information for the current view, or window
VCBlock         .Equ    VISync+1

VCLFormat       .Equ    VCBlock         ; Standard for view--Initial: Left
VCVFormat       .Equ    VCLFormat+1     ; Standard:  Initial: Appro., 0 length
VCTLVer         .Equ    VCVFormat+2     ; Top left corner, ver on screen
VCTLHor         .Equ    VCTLVer+1       ; Top left corner, hor on screen

VCTTLRow        .Equ    VCTLHor+1       ; Titles, top left row (may be zero)
VCTTLCol        .Equ    VCTTLRow+2      ; Titles, top left col
VCTLastRow      .Equ    VCTTLCol+1
VCTLastCol      .Equ    VCTLastRow+2

VCBTLRow        .Equ    VCTLastCol+1    ; Body, top left row
VCBTLCol        .Equ    VCBTLRow+2      ; Body, top left col
VCBTLVer        .Equ    VCBTLCol+1      ; Body, top left ver (of first row)
VCBTLHor        .Equ    VCBTLVer+1      ; Body, top left hor (of first col)

VCBBRRow        .Equ    VCBTLHor+1      ; Body, bottom right row
VCBBRCol        .Equ    VCBBRRow+2      ; Body, bottom right col
VCBBRVer        .Equ    VCBBRCol+1      ; Ver coord of bottom right corner of
box
VCBBRHor        .Equ    VCBBRVer+1      ; Hor coord.
VCBWidth        .Equ    VCBBRHor+1      ; Actual screen cols needed for
displayed data
VCRPartial      .Equ    VCBWidth+1      ; Boolean--right col is partial
VCTitles        .Equ    VCRPartial+1    ; TOP: x'80', left side: x'40'

VCBlockE        .Equ    VCTitles+1
; End of View (current) parameters
```

```
; Information for the secondary view (ie where the cursor isn't)
VSBlock         .Equ    VCBlockE

VSLFormat       .Equ    VSBlock         ; Initial: Left
VSVFormat       .Equ    VSLFormat+1     ; Initial: Appro., 0 length
VSTLVer         .Equ    VSVFormat+2     ; Top left corner, ver on screen
VSTLHor         .Equ    VSTLVer+1

VSTTLRow        .Equ    VSTLHor+1
VSTTLCol        .Equ    VSTTLRow+2
VSTLastRow      .Equ    VSTTLCol+1
VSTLastCol      .Equ    VSTLastRow+2

VSBTLRow        .Equ    VSTLastCol+1
VSBTLCol        .Equ    VSBTLRow+2
VSBTLVer        .Equ    VSBTLCol+1
VSBTLHor        .Equ    VSBTLVer+1

VSBBRRow        .Equ    VSBTLHor+1
VSBBRCol        .Equ    VSBBRRow+2
VSBBRVer        .Equ    VSBBRCol+1
VSBBRHor        .Equ    VSBBRVer+1
VSBWidth        .Equ    VSBBRHor+1
VSRPartial      .Equ    VSBWidth+1
VSTitles        .Equ    VSRPartial+1

VSBlockE        .Equ    VSTitles+1
; End of View (secondary) parameters


; Area to save the current window so that exact position can be
; returned to during replicate and formula expansion
SvVCBlock       .Equ    VSBlockE
SvVCCol         .Equ    VCBlockE-VCBlock+SvVcBlock
SvVCRow         .Equ    SvVcCol+1


;               More variables
Protection      .Equ    SvVCRow+2       ; Boolean, on or off
CameFrVC        .Equ    Protection+1    ; Boolean, if values need to be .
calced.

;               The following are printer formatting variables.  They
;               get initialized to zero with a new file.  The print
;               routines will initialize to real initial values.
RWidth          .Equ            CameFrVC+1      ; Of paper inches XX.X
RLMargin        .Equ            RWidth+1        ; Inches XX.X
RRMargin        .Equ            RLMargin+1      ; Inches XX.X
RChPerInch      .Equ            RRMargin+1      ; NN

RLPaper         .Equ            RChPerInch+1    ; Inches XX.X
RTMargin        .Equ            RLPaper+1       ; Inches XX.X
RBMargin        .Equ            RTMargin+1      ; Inches XX.X
RLinesPerInch   .Equ            RBMargin+1

RSpacing        .Equ            RLinesPerInch+1 ; Char S D or T, all languages
RPCC            .Equ            RSpacing+1      ; Max 13 control char for pr.
RDashes         .Equ            RPCC+14.        ; Boolean print dashes for
empty
RPrTitle        .Equ            RDashes+1       ; Print title block on report

;               Keep track of zoomed in or not
ZoomInSw        .Equ            RPrTitle+1
FCFEnable       .Equ            ZoomInSw+1      ; New 2.1
```

-- continued next page

-- continued previous page

```
;                   Work area to store record pointers.  999 for either machine
ARecords            .Equ            SSHeader+300.

BotRtRow            .Equ            ARecords+MaxSSRows+MaxSSRows+2
BotRtCol            .Equ            BotRtRow+2

ConvError           .Equ            BotRtCol+1
ConvMore            .Equ            ConvError+1
CRSPT               .Equ            ConvMore+1
CXISaveX            .Equ            CRSPT+256.
CXISWork            .Equ            CXISaveX+1
LJColNo             .Equ            CXISWork+128.
LJRowNo             .Equ            LJColNo+3.
LJCellID            .Equ            LJRowNo+4
LocFormat           .Equ            LJCellID+6

WorkType            .Equ            LocFormat+1     ; two bytes.  Values and
;                                    propd labels use 1, reg labels use 1

WorkFPD             .Equ            WorkType+2      ; Values only
WorkVal             .Equ            WorkFPD+8       ; Values and non-prop labels

CX2IResult          .Equ            WorkVal+128.
CX2IRelative        .Equ            CX2IResult+1
PrevDAddr           .Equ            CX2IRelative+1

;                   Stuff for highlighted box
BlAnchCol           .Equ            PrevDAddr+2     ; Cursor posn at entry
BlAnchRow           .Equ            BlAnchCol+1

;                   SetCorners places the top left, and bottom right vvalues
BlSoStCol           .Equ            BlAnchRow+2     ; Top left corner
BlSoStRow           .Equ            BlSoStCol+1
BlSoOppCol          .Equ            BlSoStRow+2     ; Bot right corner
BlSoOppRow          .Equ            BlSoOppCol+1

;                   Destination row and column
BlDsStCol           .Equ            BlSoOppRow+2
BlDsStRow           .Equ            BlDsStCol+1
BlDsOppCol          .Equ            BlDsStRow+2     ; Bot right corner
BlDsOppRow          .Equ            BlDsOppCol+1

;                   Note that source and destination are switched during
;                   parts of Replicate.

ReplWork            .Equ            BlDsOppRow+2    ; 128. bytes
FFData              .Equ            ReplWork+128.   ; 30 bytes
FFType              .Equ            FFData+30.
FFCoor              .Equ            FFType+1
SSCaller            .Equ            FFCoor+6        ; Command that called segment
;                   CXIPType is
;                   0:  Left paren was not following an @ function
;                   non zero:  Is the parameters byte from the @functin table
CXIPType            .Equ            SSCaller+1      ; 20. bytes, just for EditM

; This next work area is for reading and writing whole rows.  The
; size is 1550 bytes on 128K system.
BldRecWA            .Equ            CXIPType+24.

                    .If             BldRecWA+SmSSBA+1>IIMainBase
                    Crappity ss/common-sm
                    .Endc

                    .If             BldRecWA+LrgSSBA+1>PathArea
                    Crappity ss/common-lrg
                    .Endc
```

-- end of file.3B

--- FILE.36 ---

LAN 2.1 Word Processor Common Work Area

```
; Common routines for Word Processor assembly

; Equates for Word Processor
FirstDLine      .Equ            2
IntReturn       .Equ            0D0             ; Internal rep of Return
MainSegNo       .Equ            16.             ; Main is seg.16
Task            .Equ            "W"             ; ie., Quick File
VisibleCR       .Equ            7F              ; End of line in zoom in


; Zero page addresses for Quick Writer
CurSvSw         .Equ    ZP+0A8                  ; Boolean, cursor has to be found
LastVer         .Equ    ZP+0AA                  ; Last screen line available for text.
                                                ; Usually 21.
; The next two indicate what screen lines are to be displayed by routines.
DispFrom        .Equ    ZP+0AB
DispTo          .Equ    ZP+0AC
DispOver        .Equ    ZP+0AD                  ; Boolean.  Must display over previous
                                                ; data, whatever it was.
DispLn22        .Equ    ZP+0AE                  ; Boolean
DispLn23        .Equ    ZP+0AF

; The next are pointers within the data structures
CurAdDSP        .Equ    ZP+0B0                  ; Address of DSP corresponding to
                                                ; current cursor location.  Not
                                                ; always current
PrevAdDSP       .Equ    ZP+0B2

;               Zero page information about current data line
;               First for text lines
DLStickies      .Equ    ZP+0B4
DLLength        .Equ    ZP+0B5                  ; 01-50 is text
DLCRBit         .Equ    ZP+0B6

;               In case data line is command or carriage return line
DLCount         .Equ    ZP+0B4
DLCommand       .Equ    ZP+0B5      ; D0+ is command


ComBold         .Equ            2               ; Control B
ComCopy         .Equ            "C"+0AKey
ComDelete       .Equ            "D"+0AKey
ComFind         .Equ            "F"+0AKey
ComCalc         .Equ            "K"+0AKey
ComMove         .Equ            "M"+0AKey
ComChName       .Equ            "N"+0AKey
ComPrOptions    .Equ            "O"+0AKey
ComPrint        .Equ            "P"+0AKey
ComReplace      .Equ            "R"+0AKey
ComSticky       .Equ            " "+0AKey
ComTabs         .Equ            "T"+0AKey
ComUnderLine    .Equ            12.             ; Control L
ComZoom         .Equ            "Z"+0AKey
ComPageBack     .Equ            11.+0AKey
ComPageForw     .Equ            10.+0AKey
```

```
; Equates that define the printer options.  Makes them easier to change.
POPW            .Equ            0D8
POLM            .Equ            0D9
PORM            .Equ            0DA
POCI            .Equ            0DB
POP1            .Equ            0DC
POP2            .Equ            0DD
POIN            .Equ            0DE
POJU            .Equ            0DF
POUJ            .Equ            0E0
POCN            .Equ            0E1
POPL            .Equ            0E2
POTM            .Equ            0E3
POBM            .Equ            0E4
POLI            .Equ            0E5
POSS            .Equ            0E6             ; Single spacing
PODS            .Equ            0E7
POTS            .Equ            0E8
PONP            .Equ            0E9
POGB            .Equ            0EA
POGE            .Equ            0EB             ; Group.
POHE            .Equ            0EC
POFO            .Equ            0ED             ; Footer
POSK            .Equ            0EE
POPN            .Equ            0EF
POPE            .Equ            0F0
POPH            .Equ            0F1
POSM            .Equ            0F2

;               These are not available on options menu.  They are inserted
;               when printing
POPNPlus        .Equ            0F3
POPC1           .Equ            0F4     ; Page break, count is page number
POPC1Plus       .Equ            0F5     ; Page break, count+256 is page number

POPC2           .Equ            0F6     ; Page break, split a paragraph
POPC2Plus       .Equ            0F7     ; Page break, split a paragraph


;               Imbedded commands
POBB            .Equ            001
POBE            .Equ            002
POPlusB         .Equ            003
POPlusE         .Equ            004
POMinusB        .Equ            005
POMinusE        .Equ            006
POUB            .Equ            007
POUE            .Equ            008
POPP            .EQU            009
POEK            .Equ            00A
POStick         .Equ            00B
POMM            .Equ            00C     ; Out of order

; The following is the layout of the Quick Writer work area.  The beginning
; address is somewhat variable until this settles down.
                .List

; Information that is built while a file is open, and not important
; to save when closed

;               Leave a two byte buffer for mistakes
BldRecWA        .Equ            WPWA
BldRecText      .Equ            BldRecWA+1      ; Text portion

; One switch for each of first 22 lines on screen 0..21.  The value
; is the number of characters displayed the last time the line was done.
; Values are from 1 to decimal 80.  $Hex 80 is a special case of a
; carriage return display
PrevDisp        .Equ            BldRecWA+100.

; A pointer to the DSP in DSPTab for each line on the screen.  Zeroes are valid,
; and indicate beyond the end of the document
ScrAdDSP        .Equ            PrevDisp+24.
```

—— continued next page

-- continued previous page

```
; Information about previous line.  Only used occasionally
;                       First for text lines
PLStickies      .Equ            ScrAdDSP+48.
PLLength        .Equ            PLStickies+1
PLCRBit         .Equ            PLLength+1
PLRecWA         .Equ            PLCRBit+1       ; Length 82

;                       In case data line is command or carriage return line
PLCount         .Equ            PLStickies
PLCommand       .Equ            PLCount+1

; How to find the cursor if its place is marked as so many bytes
; past the first byte of a certain DSP.
CurSvDSP        .Equ            PLRecWA+82.
CurSvCount      .Equ            CurSvDSP+2      ; 3 bytes bytes to skip past.
CurSvLM         .Equ            CurSvCount+3    ; In case forget, comm lines
only
CurSvPE         .Equ            CurSvLM+1       ; Bytes past last valid char.
CurSvAtLM       .Equ            CurSvPE+1       ; Boolean, was on LM at time
CurSvVer        .Equ            CurSvAtLM+1     ; Ver when key was pressed

; Request being made of segment.  Like #ComPrint
CallerComm      .Equ            CurSvVer+1

; Tell PutCurData to bitch about lost text.  Don't in Copy.
PCBitch         .Equ            CallerComm+1

; Beginning of the word processor master record.
WPHeader        .Equ            PCBitch+1
MSNextDSP       .Equ            WPHeader
IWFiller1       .Equ            MSNextDSP+2     ; ADDSP top line when left
TabStLen        .Equ            IWFiller1+2
TabStops        .Equ            TabStLen+1      ; "-" is no-stop, "!" is stop
ZoomInSw        .Equ            TabStops+80.

; These parameters are used to return to the same screen contents whenever
; this file is selected from among those on the desktop.  Although saved
; on disk, they are re-initialized when a file is read from disk.
ResumeDSP       .Equ            ZoomInSw+1      ; Top line of screen
IWLastVer       .Equ            ResumeDSP+2
IWLastHor       .Equ            IWLastVer+1

;                       Now some miscellaneous stuff
ValidPagin      .Equ            IWLastHor+1     ; Boolean, pagination is OK
MinLM           .Equ            ValidPagin+1    ; Min Left Mar whole doc.
MMInDoc         .Equ            MinLM+1         ; Boolean, mail merge in file

;                       Work area to store record pointers.  Values higher than
;                       D000 are commands.  Variable number of records,
;                       depending on whether Slinky here or not.
;                       First text line is right at ARecords
ARecords        .Equ            WPHeader+300.

                .If             ARecords+SmWPRecs+SmWPRecs+1>IIMainBase
                Crappity
                .Endc

                .If             ARecords+LrgWPRecs+LrgWPRecs+1>PathArea
                Crappity
                .Endc
```

-- end of file.36

--- FILE.24 ---

2.1 LAN Printer Definitions Changes

```
; This becomes segment Seg.PR.  It is definitions of printers.  All of the
; addresses are relative to zero, so all word addresses will have to be
; relocated when loaded.

                .Include        /p/iall/common
                .Include        /p/iall/Macros


                .List
                .Proc           PRDefs
                .Org            0


; So other segments can get past the environment record on disk.
; Start out with enough bytes to store environment record.
                .Byte           0E0     ; HardPathName
                .Block          32.     ; Skip pathname V2.0
                .Block          4       ; Various

                .If             Language="A"
                .String         "10/01/87"
                .Endc

                .Block          10.             ; Filler
                .Block          ERELength
                .Block          ERELength
                .Block          ERELength

                .String         "Drive 2"       ; MainPathName
                .Block          50.-8           ; Remainder of MainPathName

;               Now space for 768 bytes that come from D100 thru
;               D3FF when we load.
                .Block          300


                .Include        /p/iall/segpr1

; Table of number of dots for Apple DMP proportional characters.

                .If             Language="A"
PATWidths       .Byte           07.+80,07.+80,10.,,14.          ; 20-23
                .Byte           12.,16.+80,13.,07.              ; 24-27
                .Byte           07.,07.,12.,12.                 ; 28-2B
                .Byte           07.+80,12.,07.+80,12.           ; 2C-2F

                .Byte           12.,12.,12.,12.                 ; 30-33
                .Byte           12.,12.,12.,12.                 ; 34-37
                .Byte           12.,12.,07.+80,07.+80           ; 38-3B
                .Byte           12.,12.,12.,12.+80              ; 3C-3F

                .Byte           14.,16.,15.,14.,15.,15.,15.,14. ; 40-47
                .Byte           15.,09.,13.,12.,13.,17.,16.,15. ; 48-4F

                .Byte           13.,16.,15.,12.,14.,15.,16.,17. ; 50-57
                .Byte           11.,14.,11.,12.,12.,12.,12.,17. ; 58-5F

                .Byte           07.,12.,12.,10.,12.,12.,10.,12. ; 60-67
                .Byte           12.,08.,07.,10.,08.,16.,12.,12. ; 68-6F

                .Byte           12.,12.,10.,12.,10.,12.,12.,16. ; 70-77
                .Byte           12.,12.,10.,10.,07.,10.,13.,00. ; 78-7F
                .Endc
```

```
; Table of number of 120ths for Apple Letter Quality proportional characters.

; 80 means it is punctuation.
PCTWidths       .Byte           06.+80,10.+80,10.,10.           ; 20-23
                .Byte           10.,10.+80,12.,04.              ; 24-27
                .Byte           10.,10.,10.,10.                 ; 28-2B
                .Byte           10.+80,10.,10.+80,10.           ; 2C-2F

                .Byte           10.,10.,10.,10.                 ; 30-33
                .Byte           10.,10.,10.,10.                 ; 34-37
                .Byte           10.,10.,10.+80,10.,+80          ; 38-3B
                .Byte           12.,10.,12.,10.+80              ; 3C-3F

                .Byte           10.,14.,14.,14.,14.,12.,12.,14. ; 40-47
                .Byte           14.,08.,10.,14.,12.,14.,14.,14. ; 48-4F

                .Byte           12.,14.,14.,12.,14.,14.,14.,14. ; 50-57
                .Byte           14.,14.,12.,10.,10.,10.,16.,12. ; 58-5F

                .Byte           10.,10.,12.,10.,12.,10.,08.,12. ; 60-67
                .Byte           12.,06.,06.,12.,06.,14.,12.,10. ; 68-6F

                .Byte           12.,12.,10.,10.,08.,12.,12.,14. ; 70-77
                .Byte           12.,12.,10.,10.,12.,08.,10.,10. ; 78-7F

; Table of number of dots for Epson FX proportional characters.
PFTWidths       .Byte           12.+80,05.+80,08.,12.           ; 20-23
                .Byte           12.,12.+80,12.,05.              ; 24-27
                .Byte           06.,06.,12.,12.                 ; 28-2B
                .Byte           07.+80,12.,06.+80,10.           ; 2C-2F

                .Byte           12.,08.,12.,12.                 ; 30-33
                .Byte           12.,12.,12.,12.                 ; 34-37
                .Byte           12.,12.,06.+80,06.,+80          ; 38-3B
                .Byte           16.,12.,10.,12.+80              ; 3C-3F

                .Byte           12.,12.,12.,12.,12.,12.,12.,12. ; 40-47
                .Byte           12.,08.,11.,12.,12.,12.,12.,12. ; 48-4F

                .Byte           12.,12.,12.,12.,12.,12.,12.,12. ; 50-57
                .Byte           10.,12.,10.,08.,10.,08.,12.,12. ; 58-5F

                .Byte           05.,12.,11.,11.,11.,12.,10.,11. ; 60-67
                .Byte           11.,08.,09.,10.,08.,12.,11.,12. ; 68-6F

                .Byte           11.,11.,11.,12.,11.,12.,12.,12. ; 70-77
                .Byte           10.,12.,10.,09.,05.,09.,12.,00. ; 78-7F



VeryEnd         .Equ            *

                .End

-- end of file.24
```